

Level Set Method for Simulating the Dynamics of the Fluid-Fluid Interfaces: Application of a Discontinuous Galerkin Method

Vom Fachbereich Maschinenbau
an der Technischen Universität Darmstadt

zur
Erlangung des Grades eines Doktor-Ingenieurs (Dr.-Ing.)
genehmigte

D i s s e r t a t i o n

vorgelegt von

M.Sc. Roozbeh Mousavi

aus Tehran, Iran

Berichterstatter:	Prof. Dr.-Ing.habil. M. Oberlack
Mitberichterstatter:	Prof. Dr.-Ing.habil. J. Janicka
Tag der Einreichung:	10.12.2013
Tag der mündlichen Prüfung:	29.01.2014

Darmstadt, 2014

D17

Erklärung

Hiermit erkläre ich, dass ich die vorliegende Arbeit, abgesehen von den in ihr ausdrücklich genannten Hilfen, selbstständig verfasst habe.

Roozbeh Mousavi

Bitte zitieren Sie dieses Dokument als:

URN: : urn:nbn:de:tuda-tuprints-38722

URL: <http://tuprints.ulb.tu-darmstadt.de/3872/>

Dieses Dokument wird bereitgestellt von tuprints, E-Publishing-Service der TU Darmstadt.

<http://tuprints.ulb.tu-darmstadt.de>

tuprints@ulb.tu-darmstadt.de

Die Veröffentlichung steht unter folgender Creative Commons Lizenz:

Namensnennung-Keine kommerzielle Nutzung-Keine Bearbeitung 2.0 Deutschland

<http://creativecommons.org/licenses/by-nc-nd/2.0/de/>



To my lovely daughter Mana

Abstract

A discontinuous Galerkin (DG) method was applied for simulating the dynamics of the fluid-fluid interfaces. The numerical implementations were performed in the context of an available in-house code BoSSS, see (Kummer 2012). The flow field was assumed to be governed by a single-set of the Navier-Stokes equation in terms of the phase-dependent density and viscosity fields. As in the discontinuous Galerkin method the variables in each cell are expressed in terms of a polynomial space, the solution may exhibit spurious oscillations in the presence of the steep variations such as the density jumps across the interface. In order to overcome this problem, a diffuse interface assumption was made, according to which a jump is approximated by a continuous variation employing a regularized heaviside function. The interface diffusion is supposed to take place in a region with a reasonable width. Therefore, in order to properly express the smoothed jumps in terms of a polynomial space of a certain degree, only jumps with limited height could be considered. Otherwise, the grid needs to be highly refined in the interface diffusion region for preventing the non-physical spatial oscillation of the solution. Surface tension effects as well as gravity were also involved in the simulations by adding the corresponding source terms to the Navier-Stokes equation. The interface kinematics was simulated using the level set method. Taking the advantage of the discontinuous Galerkin method, a precise solution to the level set advection equation was achieved. As the regularized Heaviside and delta functions are commonly expressed in terms of the level set function, the level set function needs to remain signed distance in order to keep a uniform diffusion width. The signed distance property of a level set functions was recovered by solving the re-initialization equation. A Godunov's scheme was applied for approximating the Hamiltonian of the re-initialization equation, in order to obtain a solution with a monotonicity preserving behavior. A notable stability improvement was achieved by adding an artificial diffusion along the characteristic lines of the re-initialization equation. The solution showed an appropriate hp-convergence behavior and almost no spurious movement of the interface was detected. For solving the Navier-Stokes equation, an explicit-implicit stiffly stable time integration method was employed combined by a splitting method for decoupling the velocity and pressure fields within the DG framework. This solver which had been priorly implemented by Emamy (2013) for the single phase formulation of the equation, was used as a basis for implementing a new solver for the multiphase formulation. The multiphase flow solver was verified by considering a number of the test cases, such as a rising bubble.

Zusammenfassung

In der vorliegenden Arbeit wird eine diskontinuierliche Galerkin Methode zur numerischen Simulation der Dynamik von Fluid Fluid Interfaces verwendet.

Zur Durchführung der Simulationen wurde der institutseigene Code BoSSS erweitert, siehe (Kummer 2012). Dabei wird angenommen, dass das Strömungsfeld durch die Navier-Stokes Gleichungen mit phasenabhängigen Dichte und Viskositätsfeldern beschrieben werden. Im Rahmen der diskontinuierliche Galerkin Methode werden die abhängigen Variablen durch polynomiale Funktionen in jeder Zelle des numerischen Gitters approximiert. Dieser Ansatz führt in der Regel zu unphysikalischen Oszillationen an der Phasengrenzfläche, da dort Sprünge in den physikalischen Größen wie beispielsweise der Dichte vorliegen. Um dieses Problem zu überwinden, wurde ein diffuses Interface angenommen, bei der ein Sprung durch eine kontinuierliche Variation, d.h. eine geglättete Heavisidefunktion, approximiert wird. Außerdem wird angenommen, dass die Interfacediffusion in einem begrenzten Bereich um die Phasengrenzfläche stattfindet. Bei gegebener Gitterauflösung und Polynomordnung können nur Sprünge mit einer begrenzten Sprunghöhe betrachtet werden, um die genannten unphysikalischen Oszillationen zu vermeiden. Effekte aus Oberflächenspannung und Gravitation werden in der Simulation durch Hinzufügen der entsprechenden Quellterme in den Navier-Stokes Gleichungen berücksichtigt. Zur Beschreibung der Kinematik der Phasengrenzfläche wird die Level Set Methode verwendet. Durch Verwendung der diskontinuierlichen Galerkin Methode zur Lösung der Level Set Advektionsgleichung können Ergebnisse mit hoher Genauigkeit erreicht werden. Da die geglättete Heavisidefunktion sowie die Deltafunktion in Abhängigkeit der Level Set Funktion beschrieben werden, muss es sich bei der Level Set Funktion um eine vorzeichenbehaftete Abstandsfunktion handeln, um eine gleichförmige Diffusionsweite zu erhalten. Die Eigenschaft eines vorzeichenbehafteten Abstands der Level Set Funktion kann durch Lösen der Reinitialisierungsgleichung wiederhergestellt werden. Die Anwendung eines Godunov Schema zur Approximation des Hamiltonterms in der Reinitialisierungsgleichung führt zu einer monotonie erhaltenden Lösung. Eine erwähnenswerte Verbesserung der Stabilität wird durch Hinzufügen einer künstlichen Diffusion entlang der charakteristischen Linien erreicht. Die Lösung zeigt das erwartete hp Konvergenz Verhalten und nahezu keine unphysikalische Verschiebung der Phasengrenzfläche werden beobachtet. Zur Lösung der Navier-Stokes Gleichungen wird eine explizit implizite, steife Integrationsmethode in Kombination mit einer Splittingmethode zur Entkopplung des Geschwindigkeits und des Druckfelds verwendet. Dieser Solver, welcher ursprünglich zur Simulation von Einphasenströmungen entwickelt wurde, siehe (Emamy 2013), diente als Basis zur Implementierung des neuen Mehrphasenlösers. Dieser neue Löser für Mehrphasenströmungen wird anhand verschiedener Testfälle wie der einer aufsteigenden Blase verifiziert.

Acknowledgments

This research was accomplished in the Chair of Fluid Dynamics of the Mechanical Engineering Department in Technische Universität Darmstadt. The work was financially supported by the Center of Smart Interfaces in TU Darmstadt.

I would like to thank Prof. Dr. -Ing. Habil. Martin Oberlack for admitting me as a PhD student in the Chair of Fluid Dynamics. In addition, I would like to thank Prof. Dr. -Ing. Habil. Johannes Janicka for accepting to be my second supervisor.

Concerning the research, my greatest thanks goes to Dr. -Ing. Nehzat Emamy for her valuable scientific support. I highly benefited from her deep knowledge in fluid mechanics and numerical analysis and her outstanding skills in computer programming.

In addition, I would like to thank Dr. -Ing. Florian Kummer, Dr. -Ing. Björn Müller, Dipl. -Ing. Benedikt Klein, Prof. Dr. -Ing. Habil. Martin Oberlack and Prof. Dr. -Ing. Habil. Yongqi Wang for fruitful discussions. Moreover, I would like to thank Dr. -Ing. Florian Kummer especially for technical supports in using the code BoSSS.

Concerning translating the abstract of this dissertation to German, I would like to thank Dipl. -Math. -Ing. Andreas Zieleniewicz and Dipl. -Ing. Benedikt Klein for their kind supports.

I would like also to mention my professors in Shiraz University where I did my Bachelor's and Master's studies. I would like to thank Prof. Dr. Ghodrat Karami, Prof. Dr. Mohammad M. Alishahi and Prof. Dr. Homayoun Emdad, who had major roles in constructing the foundation of my mechanical engineering knowledge.

At this point when I am finalizing my academic education, I would like to dedicate my deepest thanks to my father, Mr. Sadegh Mousavi, and my mother Mrs. Zahra Sanati. My father always motivated me to perform scientific researches and my mother was always careful about my studies and performance at school.

I would like to thank my wife Dr. -Ing. Nehzat Emamy again for her continued support.

Contents

Nomenclature	xiii
1 Introduction	1
1.1 Motivations	1
1.2 Objectives	3
1.3 Structure of the Dissertation	3
2 Modeling the Dynamics of Fluid-Fluid Interfaces	4
2.1 Level Set Method for Modeling Moving Interfaces	4
2.2 Two-phase Flow Navier-Stokes Equations	5
3 Numerical Methodology	9
3.1 An Overview on the Numerical Techniques for Simulating the Inter- face Kinematics	9
3.2 Signed Distance Property of the Level Set Functions	11
3.3 Higher-Order Numerical Schemes for Solving the Governing Equations	14
3.3.1 Spatial Discretization Methods	15
3.3.1.1 Finite Difference Method	16
3.3.1.2 Finite Volume Method	18
3.3.1.3 Spectral Methods	19
3.3.1.3.1 Method of Weighted Residuals	20
3.3.1.4 Numerical integration	22
3.3.2 Temporal Discretization Methods	22
3.3.2.1 Linear Multi-Step Methods	23
3.3.2.2 Runge-Kutta Methods	23
3.4 Discontinuous Galerkin Method	25
3.4.1 Solution Representation	25
3.4.1.1 Orthonormal Basis Polynomial Space	27
3.4.1.2 Solution Discontinuity	28
3.4.2 Spatial Discretization	29
3.4.3 Gradient Calculation	33
3.4.4 Boundary Condition	34
3.4.5 Reference Cell	34
3.4.6 CFL Condition	36
3.5 The In-House Code "BoSSS"	36
3.6 Solving the Level Set Advection Equation	37
3.7 Solving the Level Set Re-Initialization Equation	37
3.7.1 Godunov's Scheme	38

3.7.2	Godunov's Scheme for Solving the Level Set Re-initialization Equation	39
3.8	Solving the Multiphase Formulation of the Navier-Stokes Equation	40
4	Numerical Simulations and Results	43
4.1	Error Calculation	43
4.1.1	Volume/Area Loss	43
4.1.2	Interface L^1 -Error	43
4.1.3	L^2 Error	44
4.2	Visualization	44
4.3	Solutions to the Level Set Advection Equation	46
4.3.1	Rigid Body Rotation of a Circle	46
4.3.2	Rigid Body Rotation of a Slotted Disk	56
4.3.3	Periodic Swirling of a Circle	60
4.3.4	Deformation of a Circle in a Channel Flow	73
4.3.5	Periodic Swirling of a Slotted Disk	77
4.3.6	Deformation of a Circle in a Multi-Vortex Flow	80
4.3.7	Periodic Swirling of a Sphere	86
4.4	Solutions to the Level Set Re-Initialization Equation	92
4.4.1	Re-initializing the Level Set Function of a Circle	92
4.4.2	Re-initializing the Level Set Function of an Arc	106
4.4.3	Re-initializing the Level Set Function of an Ellipse	108
4.4.4	Re-initializing the Level Set Function of a Deformed Circle in a Channel Flow	110
4.5	Solutions to the Multiphase Formulation of the Navier-Stokes Equation	112
4.5.1	Capillary Pressure and the Spurious Currents	112
4.5.2	Capillary force Exerted on a Square Bubble	122
4.5.3	Capillary force Exerted on an Elliptic Bubble	126
4.5.4	Rising of a Circular Bubble	132
5	Conclusion	138
6	Bibliography	141
	Appendix A	145

Nomenclature

$\delta(\phi)$	Dirac delta function
δ_{ji}	Kronecker delta function
ϵ	Smoothing width of the soothed Heaviside functions
$\hat{\varphi}_{jk}$	Corresponding coefficient of the j^{th} basis polynomial over the k^{th} cell
κ	Interface mean curvature
$\llbracket \varphi_{hp} \rrbracket$	Jump of φ_{hp} across the cell border $\partial\Omega_{h,k}$
\mathbf{A}_k	Geometrical Translation matrix
\mathbf{D}	Rate of stress tensor
\mathbf{D}_i	Rate of stress tensor within the phase i
\mathbf{e}_i	i^{th} standard basis
\mathbf{f}^*	Numerical flux
\mathbf{G}	Gradient vector
\mathbf{g}	Vector of the gravity acceleration
\mathbf{M}_k	Geometrical deformation matrix
\mathbf{n}	Interface normal vector
\mathbf{n}_s	Normal vector to the border of a cell
\mathbf{T}_k	Corresponding transformation operator of k^{th} cell between ξ and \mathbf{x} spaces
\mathbf{u}	Velocity vector field
\mathbf{u}_i	Velocity vector field within the phase i
$\mathbf{u}_{\mathcal{I}}$	Velocity of the interface \mathcal{I}
$\mathbf{x}_{\mathcal{I}}(t)$	Position of the interface \mathcal{I}
$\mathcal{H}(\phi)$	Heaviside function
\mathcal{I}	A fluid-fluid interface
$\mathcal{R}(\tilde{\varphi})$	Residual term as a result of approximating φ by $\tilde{\varphi}$
μ_i	Viscosity within the phase i
Ω_h	A discrete domain
$\Omega_{h,k}$	k^{th} numerical cell
$\partial\Omega_h$	boundary of the discrete domain Ω_h

$\partial\Omega_{h,k}$	Border of the cell $\Omega_{h,k}$
ϕ	level set function
ϕ^{ref}	reference level set function
ρ_i	Density within the phase i
σ	Surface tension
τ	Stress tensor
τ_i	Stress tensor within the phase i
$\tilde{\varphi}^n$	Approximation of φ at time step n
φ_B	Dirichlet boundary condition for φ
$\varphi_{hp,k}$	DG-based representation of φ (DG field of φ)
φ_{hp}^+	Outer-cell value of φ_{hp} at the border of a cell
φ_{hp}^-	Inner-cell value of φ_{hp} at the border of a cell
ϑ_j	Space of the basis functions (space of the orthogonal polynomials)
$d(\mathbf{x}, t)$	distance to the interface \mathcal{I}
N_C	Number of the cells
N_P	Dimension of the space of the orthogonal basis polynomials
N_{DoF}	Number of numerical degrees of freedom
P	Pressure field
p	Degree of a DG field
P_i	Pressure field within the phase i
s_φ	Source term
1D	One-dimensional
2D	Two-dimensional
3D	Three-dimensional
AM	Advection Matrix
DM	Diffusion matrix
SM	Source matrix
TM	Temporal matrix
CFL	Courant-Friedrichs-Lewy
DG	Discontinuous Galerkin
EEO	Experimental Error Order
FD	Finite difference
FE	Finite Element

FS	Fourier spectral
FV	Finite volume
HJ	Hamilton-Jacobi
LMS	Linear multi-step
LS	level set
LSA	level set advection
LSRI	level set re-initialization
NS	Navier-Stokes
NSDLS	non- signed distance level set
OBPS	orthonormal basis polynomial space
RK	Runge-Kutta
SDLS	signed distance level set
TVD	Total variation diminishing

1 Introduction

1.1 Motivations

The flows consisting of fluids with different properties, are termed multiphase flows. Despite the name of this kind of the flows, the constitutive fluids commonly do not need to be thermodynamically in different phases. If the fluids are immiscible, they are separated by thin layers termed interface. These layers are the regions across which, the fluid properties as well as some of the flow variables are subjected to steep variations. In the context of the continuum mechanics, an interface is represented as a geometrical surface with zero thickness. Making this assumption can produce a number of the mathematical singularities, see (Shikhmurzaev 2007).

Multiphase flows are ubiquitous in the nature and technology. In nature, several phenomena related to multiphase flows are employed intelligently in different ways providing brilliant ideas for engineering designs. For instance, plants use the capillary tubes in order to transfer water along long distances. This fact which is based on the concept of the surface tension, is also used in industry, for example to transfer liquids in porous medias. Interface dynamics plays a major role in hydrodynamic designs. For instance, the body of a watercraft should be designed in a way not to generate large waves. Otherwise, the propulsion energy is spent more for generating waves than pushing water out of the way. Interface dynamics can be affected by complex flow phenomena such as turbulence. For instance, injectors of internal combustion engines employ such an effect in order to make the fuel atomization.

The few examples mentioned above, demonstrate the wide spectrum of natural phenomena as well as the industrial applications which are in contact to the concept of multiphase flows. This reflects an extensive interest for investigating the characteristics of such a kind of flows.

Engineering designs are commonly based on using authorized codes and standards such as the ones published by ASME (American Society of Mechanical Engineering). Experimental measurements are often done either for verifying the designs or for being used in the procedures of the innovative designs. As performing an experiment is usually costly, numerical simulation can be used for providing estimates and skipping unnecessary experiments. Therefore improving the accuracy of the numerical simulation result in a more realistic estimation. Moreover, numerical simulations can be used when there is a lack of proper experimental tool for measuring certain flow variables. In this case, the accuracy of the numerical simulation is very important.

The first step in making a numerical simulation is constructing a consistent mathematical model comprising a set of governing equations. The second step is employing suitable numerical methods for obtaining accurate solutions to these equations. A major issue in numerical simulation of multiphase flows is the manner of repre-

senting the interface and simulating its kinematics. Several methods with different levels of the accuracy and robustness have been proposed, among which, the level set (LS) method, see (Osher & Fedkiw 2003), is a robust one. The interface in the LS method is represented as the zero iso-value of a function which is called the LS function. The implicit representation of the interface provides an appropriate way for simulating the topological changes. Moreover, as the interface does not need to be reconstructed, this method is quite suitable when a precise calculation of the curvature is required. The accuracy of this method in simulating the interface kinematics is highly dependent on the preciseness of the numerical method applied for solving the corresponding advection equation, namely the level set advection (LSA) equation. The discontinuous Galerkin method (DG), see (Pietro & Ern 2011), is a modern technique providing a framework where a higher-order approximation can be implemented in a robust way. The variables are expressed in this method in each cell in terms of an orthonormal basis polynomial space (OBPS). As in this method in-cell variations are considered, an acceptable level of the accuracy can be attained using a rather low spatial resolution. In this way, the total number of degrees of freedom is reduced although it is increased in each cell. The research presented in (Marchandise et al. 2006) is a well-known pioneer study on applying the DG method for solving the LSA equation. According to the excellent accuracy they achieved, they claimed that DG is the best technique for solving the hyperbolic equations, such as the LSA equation. Detailed studies such as an *hp*-convergence analysis is still missing in the literature. Although in the DG-based LS method the LS function does not need to be signed distance, this property is required for making a uniform diffusion thickness when a diffuse interface assumption is made, see e.g. (Zahedi et al. 2009, Grooss & Hesthaven 2006). The signed distance property of an LS function can be recovered by solving an Eikonal equation termed as the level set re-initialization (LSRI) equation, see (Sussman et al. 1998). Application of the DG method for solving the LSRI equation has been only considered in (Grooss & Hesthaven 2006) yet. Although they have employed a set of the stabilization techniques successfully, a lack of performing a procedural error analysis is obvious in their publications.

Concerning the numerical simulation of incompressible multiphase flows, although in (Marchandise & Remacle 2006) the LSA equation is solved applying the DG method, for solving the Navier-Stokes (NS) equation a lower-order finite element method (FE) was applied. As they used different grids for the DG and FE methods, they faced difficulties in calculating the curvature as a result of the necessity of projecting the field of the LS function between the grids, see (Marchandise et al. 2007). The research presented in (Grooss & Hesthaven 2006) is so far the only study applying the DG method for both of the LS and the NS equations.

The present research is mainly motivated by the need of performing the mentioned lacks of the procedural error analyses for solving the LSA and LSRI equations. Moreover, this work is aimed to discover the major challenges inherent in the application of the DG method for simulating the multiphase flows based on the diffuse interface assumption.

1.2 Objectives

The major objectives of the present research can be listed as follows:

- Verifying the application of the DG method for solving the the LSA and LSRI equations.
- Verifying the application of the DG method for solving the multiphase formulation of the NS equation making a diffuse interface assumption.

1.3 Structure of the Dissertation

The present dissertation is organized as follows:

Chapter 2 is assigned to explaining the mathematical modeling of the dynamics of the fluid-fluid-interfaces. This chapter begins by explaining the mathematical representation of the interface using the LS method. Afterward, the derivation of the multiphase formulation of NS equation following the one-fluid approach is briefly described.

Chapter 3 is assigned to discuss the numerical methodology. This chapter begins by a section reviewing the different approaches reported in the literature for the numerical representation of the interface. As in the present research the LS method is used for simulating the interface kinematics, this section is continued by explaining more details on the LS method including the re-initialization of the LS function. The next section of this chapter is assigned to explaining the numerical methods applied for solving the governing equations of the present research. As in the present research the DG method is used for the spatial discretization of the equations, this section is aimed to provide a theoretical insight into the major principles of this method. The implementations of this research are performed in the context of an available in-house code which was initiated as the objective of another PhD project, see (Kummer 2012). Therefore, describing the lower level structure of this code is not in the scope of the present dissertation.

Chapter 4 is assigned to presenting a number of the simulations performed in the present research and discussing the results. This chapter includes three main sections presenting the solutions to the LSA equation, LSRI equation and the multiphase formulation of the NS equation, respectively.

Chapter 5 is assigned to summarizing the research as well as a set of the suggestions for the future works.

2 Modeling the Dynamics of Fluid-Fluid Interfaces

2.1 Level Set Method for Modeling Moving Interfaces

An interface in the LS method is represented implicitly as the zero-iso value of a function which is known as the LS function. Using the name "level set" is because that this function is considered as a set of the iso-values or the levels. The implicit representation provides the ability of handling any topological changes of the interface. Figure 2.1 demonstrates representation of an interface using the LS method. As it is shown

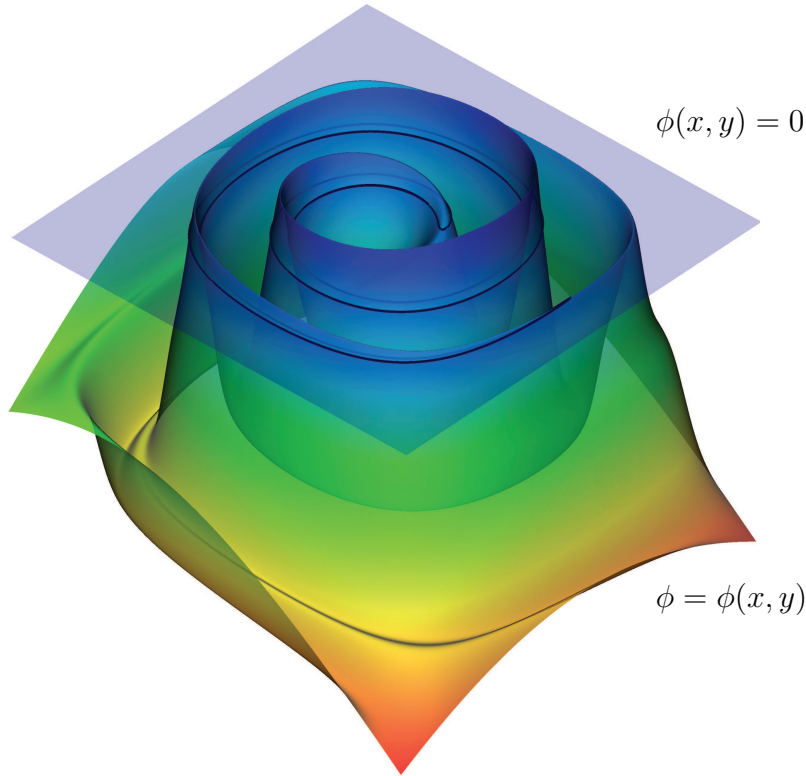


Figure 2.1 Representation of an interface as the zero iso-value of an LS function

in the picture, although the LS function is define over whole the domain, only its zero iso-value, separating the negative and positive regions of the function, is used for the interface representation. Therefore, any function that its zero iso-value represents the interface, can be used as the LS function. But the LS function is commonly designed to be a function that returns the signed-distance to the interface. The signed-distance LS function (SDLS) $\phi(\mathbf{x}, t)$ of an interface \mathcal{I} is defined as,

$$\phi(\mathbf{x}, t) = \begin{cases} -d(\mathbf{x}, t), & \phi < 0, \\ 0, & \phi = 0, \\ d(\mathbf{x}, t), & \phi > 0, \end{cases} \quad (2.1)$$

where $d(\mathbf{x}, t)$ is the distance to the interface defined as,

$$d(\mathbf{x}, t) = \min(|\mathbf{x} - \mathbf{x}_I(t)|), \quad (2.2)$$

where $\mathbf{x}_I(t)$ denotes the interface position. Parameterizing the interface with the surface coordinates (ξ_I, η_I) , the interface position can be determined by,

$$\mathbf{x}_I(t) = \mathbf{x}(\xi_I, \eta_I, t). \quad (2.3)$$

The value of the gradient of an SDLS function is uniformly equal to 1. The advantage of using an SDLS function is that this LS function can be used for constructing the distributions for which the distance to the interface is a parameter, for instance, smoothed Heaviside functions which are considered in the chapter 3.

Following (Sussman et al. 1998), evolution of the interface \mathcal{I} can be determined by,

$$\frac{d\mathbf{x}_I(t)}{dt} = \frac{\partial \mathbf{x}(\xi_I, \eta_I, t)}{\partial t} = \mathbf{u}_I(\mathbf{x}(\xi_I, \eta_I, t), t), \quad (2.4)$$

where \mathbf{u}_I denotes the interface velocity. As $\phi(\mathbf{x}(\xi_I, \eta_I, t), t)$ is defined to be zero for all the time, one can write,

$$\begin{aligned} \frac{d\phi(\mathbf{x}(\xi_I, \eta_I, t), t)}{dt} &= \frac{\partial \phi}{\partial t} + \frac{\partial \phi}{\partial x_1} \frac{\partial x_1(\xi_I, \eta_I, t)}{\partial t} + \frac{\partial \phi}{\partial x_2} \frac{\partial x_2(\xi_I, \eta_I, t)}{\partial t} + \frac{\partial \phi}{\partial x_3} \frac{\partial x_3(\xi_I, \eta_I, t)}{\partial t} \\ &= \frac{\partial \phi}{\partial t} + \frac{\partial \phi}{\partial x_{1I}} u_{1I} + \frac{\partial \phi}{\partial x_{2I}} u_{2I} + \frac{\partial \phi}{\partial x_{3I}} u_{3I} = 0, \end{aligned} \quad (2.5)$$

where x_i represents the components of \mathbf{x} , and u_{iI} represents the components of \mathbf{u}_I . Assuming the interface \mathcal{I} to be a material surface, for which, the velocity of the interface is equal to the velocity of the fluid particles located on the interface, one can write the following equation describing advection of the LS function,

$$\frac{\partial \phi}{\partial t} + \mathbf{u} \cdot \nabla \phi = 0, \quad (2.6)$$

where \mathbf{u} is the field of fluid velocity.

2.2 Two-phase Flow Navier-Stokes Equations

Considering an incompressible two-phase flow system as shown in figure 2.2, the continuity equation can be written for each of the phases as,

$$\nabla \cdot \mathbf{u}_1 = 0, \quad (2.7)$$

$$\nabla \cdot \mathbf{u}_2 = 0, \quad (2.8)$$

where the index 1 is assigned to the domain with $\phi < 0$ and the index 2 is assigned to the domain with $\phi > 0$. Moreover, \mathbf{u}_1 and \mathbf{u}_2 denote the velocities within the phases 1 and 2, respectively.

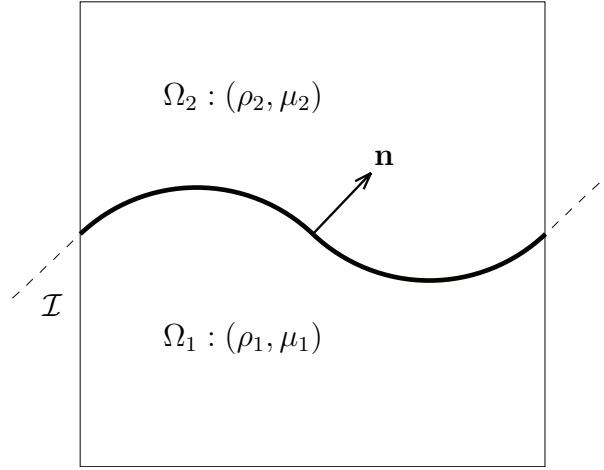


Figure 2.2 A two-phase flow system including a dynamic interface

The momentum equation can be written for each of the phases as,

$$\begin{aligned} \rho_1 \frac{D\mathbf{u}_1}{Dt} &= -\nabla P_1 + \nabla \cdot (2\mu_1 \mathbf{D}_1) + \rho_1 \mathbf{g}, \quad \text{for } \mathbf{x} \in \Omega_1 \\ \rho_2 \frac{D\mathbf{u}_2}{Dt} &= -\nabla P_2 + \nabla \cdot (2\mu_2 \mathbf{D}_2) + \rho_2 \mathbf{g}, \quad \text{for } \mathbf{x} \in \Omega_2 \end{aligned} \quad (2.9)$$

where P_1 and P_2 denote the pressures within the phases and \mathbf{D}_1 and \mathbf{D}_2 denote the rate of strain tensors, defined by,

$$\mathbf{D}_i = 1/2(\nabla \mathbf{u}_i + \nabla^T \mathbf{u}_i). \quad (2.10)$$

The densities of the phases are denoted by ρ_1 and ρ_2 and the viscosities by μ_1 and μ_2 . Vector of the gravity acceleration is denoted by \mathbf{g} .

Denoting the stress tensors in the phases 1 and 2 by τ_1 and τ_2 , defined by,

$$\tau_i = -P_i \mathbf{I} + 2\mu_i \mathbf{D}_i, \quad (2.11)$$

the following boundary condition can be imposed on the interface,

$$2(\mu_2 \mathbf{D}_2 - \mu_1 \mathbf{D}_1) \mathbf{n} = (P_2 - P_1 + \sigma \kappa) \mathbf{n}, \quad (2.12)$$

where σ denotes the surface tension, \mathbf{n}_x denotes the interface normal vector defined by

$$\mathbf{n} = \frac{\nabla \phi}{|\nabla \phi|}, \quad (2.13)$$

and κ denotes the interface mean curvature defined by,

$$\kappa = -\nabla \cdot \mathbf{n}. \quad (2.14)$$

Making a material interface assumption as,

$$\mathbf{u}_1(\mathbf{x}_x(t), t) = \mathbf{u}_2(\mathbf{x}_x(t), t) = \mathbf{u}_x(\mathbf{x}_x(t), t), \quad (2.15)$$

a single continuity equation can be written as,

$$\nabla \cdot \mathbf{u} = 0, \quad (2.16)$$

where \mathbf{u} is the velocity vector field within the entire domain of two-phase flow. In addition, following (Chang et al. 1996), the momentum equation (2.9) together with the boundary condition (2.12) can be combined to a single momentum equation as,

$$\frac{\partial \mathbf{u}}{\partial t} + \nabla \cdot (\mathbf{u}\mathbf{u}) = -\frac{\nabla P}{\rho(\phi)} + \frac{\nabla \cdot (2\mu(\phi)\mathbf{D})}{\rho(\phi)} + \frac{\sigma\kappa\mathbf{n}\delta}{\rho(\phi)} + \mathbf{g}, \quad (2.17)$$

where P and \mathbf{D} denote the fields of pressure and the rate of strain tensor within the entire domain of two-phase flow, respectively. $\delta(\phi)$ denotes the Dirac delta function defined as,

$$\delta(\phi) = \begin{cases} \infty & , \text{if } \phi = 0 \\ 0 & \text{otherwise.} \end{cases} \quad (2.18)$$

$\rho(\phi)$ and $\mu(\phi)$ denote the functions returning the density and viscosity distributions over the entire domain of the two-phase flow, respectively, which are defined as,

$$\rho(\phi) = \rho_2(1 - \mathcal{H}(\phi)) + \rho_1\mathcal{H}(\phi), \quad (2.19)$$

$$\mu(\phi) = \mu_2(1 - \mathcal{H}(\phi)) + \mu_1\mathcal{H}(\phi). \quad (2.20)$$

where $\mathcal{H}(\phi)$ denotes a Heaviside function defined as

$$\mathcal{H}(\phi) = \begin{cases} 0, & \phi \leq 0, \\ 1, & \phi > 0. \end{cases} \quad (2.21)$$

Equations (2.16) and (2.17) holds over the entire domain of the two-phase flow. A generalized formulation is derived by Wang & Oberlack (2011) for a three-phase flow system with the interfaces intersecting at a contact line. They also considered the spatial variation of the surface tension in order to model the Marangoni effects.

It is more convenient to use the dimensionless form of the NS equations which can be derived by introducing the following dimensionless variables, see e.g. (Zahedi et al. 2009),

$$\begin{aligned} \mathbf{x}^* &= \frac{\mathbf{x}}{l_{\text{ref}}}, & \mathbf{u}^* &= \frac{\mathbf{u}}{u_{\text{ref}}}, & t^* &= \frac{t}{t_{\text{ref}}}, \\ P^* &= \frac{P}{P_{\text{ref}}}, & \rho^*(\phi) &= \frac{\rho(\phi)}{\rho_{\text{ref}}(\phi)}, & \mu^*(\phi) &= \frac{\mu(\phi)}{\mu_{\text{ref}}(\phi)}, \end{aligned} \quad (2.22)$$

Substituting the new variables into the equations (2.16) and (2.17) yields,

$$\frac{u_{\text{ref}}}{l_{\text{ref}}} \nabla \cdot \mathbf{u}^* = 0 \quad (2.23)$$

$$\begin{aligned} \frac{u_{\text{ref}}}{t_{\text{ref}}} \frac{\partial \mathbf{u}^*}{\partial t^*} + \frac{u_{\text{ref}}^2}{l_{\text{ref}}} \nabla \cdot (\mathbf{u}^* \mathbf{u}^*) &= - \frac{P_{\text{ref}}}{l_{\text{ref}} \rho_{\text{ref}}} \frac{\nabla P^*}{\rho^*(\phi)} + \frac{\mu_{\text{ref}} u_{\text{ref}}}{l_{\text{ref}}^2 \rho_{\text{ref}}} \frac{\nabla \cdot (2\mu(\phi)^* \mathbf{D}^*)}{\rho^*(\phi)} \\ &+ \frac{1}{l_{\text{ref}}^2 \rho_{\text{ref}}} \frac{\sigma \kappa \mathbf{n} \delta_{\mathcal{I}}}{\rho^*(\phi)} - \mathbf{g}, \end{aligned} \quad (2.24)$$

where t_{ref} and P_{ref} can be calculated as,

$$t_{\text{ref}} = \frac{l_{\text{ref}}}{u_{\text{ref}}}, \quad P_{\text{ref}} = \rho u_{\text{ref}}^2. \quad (2.25)$$

Introducing the dimensionless parameters of Reynolds, Weber and Froude as,

$$Re = \frac{\rho_{\text{ref}} u_{\text{ref}} l_{\text{ref}}}{\mu_{\text{ref}}}, \quad We = \frac{\rho_{\text{ref}} u_{\text{ref}}^2 l_{\text{ref}}}{\sigma}, \quad Fr = \frac{u_{\text{ref}}}{\sqrt{g l_{\text{ref}}}} \quad (2.26)$$

the following dimensionless two-phase formulation of the incompressible NS equation can be obtained by performing some mathematical manipulations and omitting the (*) superscript,

$$\nabla \cdot \mathbf{u} = 0 \quad (2.27)$$

$$\nabla \cdot \mathbf{u} = 0, \quad (2.28)$$

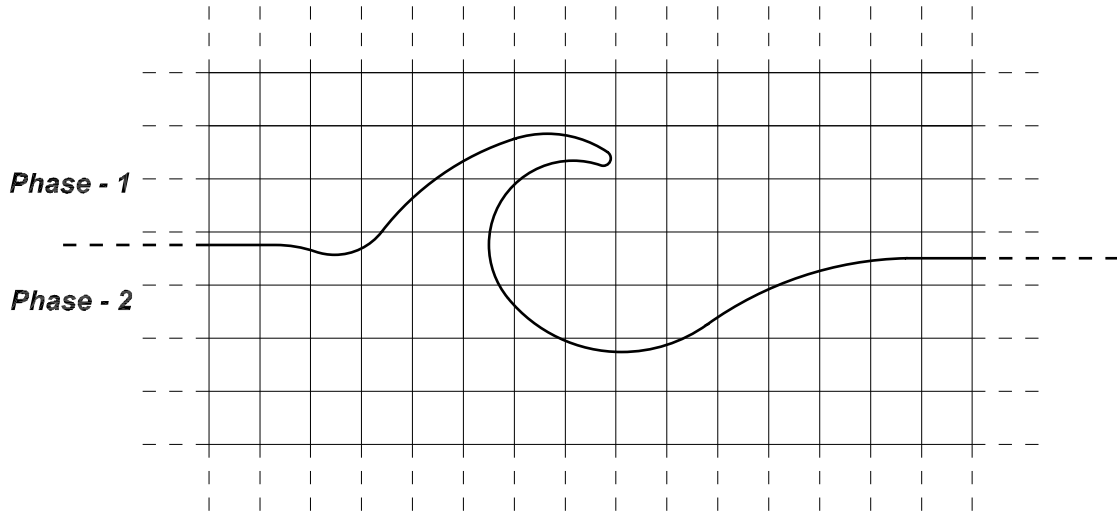
$$\frac{\partial \mathbf{u}}{\partial t} + \nabla \cdot (\mathbf{u} \mathbf{u}) = - \frac{\nabla P}{\rho(\phi)} + \frac{1}{Re} \frac{\nabla \cdot (2\mu \mathbf{D})}{\rho(\phi)} + \frac{\sigma \kappa \mathbf{n} \delta_{\mathcal{I}}}{\rho(\phi) We} + \frac{\mathbf{e}_g}{Fr^2}, \quad (2.29)$$

where \mathbf{e}_g denotes the gravity direction.

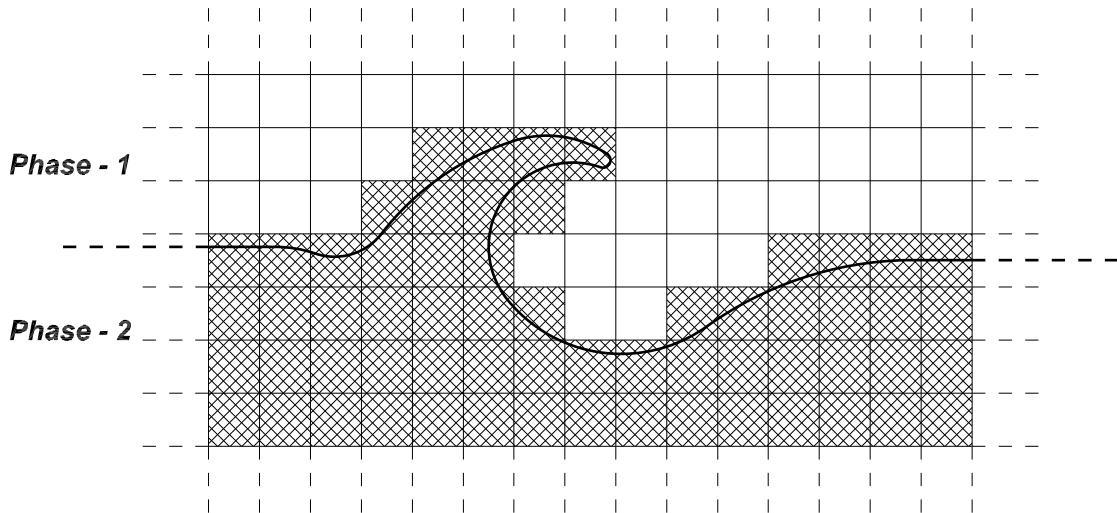
3 Numerical Methodology

3.1 An Overview on the Numerical Techniques for Simulating the Interface Kinematics

The methods proposed for the numerical representation of the interface, can be classified into two categories, namely the surface methods and the volume methods, see e.g. (Ferziger & Peric 2002). The surface methods consider the interface itself as an object, either explicitly or implicitly. Whereas, the volume methods consider the phases at the either sides of the interface. Therefore, in the volume methods in order to obtain the normal vector to the interface, the interface is required to be reconstructed. As the interface curvature which is used to predict the surface tension effects, is the divergence of the normal vector, any inaccuracy in calculating the normal vector is more heavily represented in the curvature. A common consequence of a non-precise prediction of the surface tension effects is the formation of a set of spurious vortical flows, see e.g. (Lafaurie et al. 1994). Hence, a surface method is often a recommended choice when the surface tension effects is involved in the problem. The difference between these two types of methods is schematically demonstrated in figures 3.1a and 3.1b. The front tracking (FT) method introduced by Glimm et al. (1988), as well as the LS method introduced by Osher & Sethian (1988), are the common methods which can be classified under the category of the surface methods. The FT method explicitly represents the interface by a set of the connected massless particles. The particles are advected through the domain in a Lagrangian way accompanying to a set of the conditions enforced on the interface. This method is very accurate if the interface is reconstructed by passing a order spline of higher-degree over the particles. But the necessity of tracking a rather large number of particles makes this method expensive. Moreover, as the particles need to keep an optimum distance to each other, several particles are necessary to be added or removed during the simulation. In addition, a new reconnection procedure needs to be performed after any change in the configuration of the particles. Furthermore, for the simulation of the interface breakup or coalescence an ad-hoc procedure needs to be performed, see e.g. (Unverdi & Tryggvason 1992). As it was explained in the section 2.1, the LS method implicitly represents the interface as the zero iso-value of a function which is called the LS function. It is more convenient that this function returns the signed distance to the interface. The interface kinematics is simulated by solving an advection equation for the LS function, i.e. the equation (2.6). In order to restore the signed distance property which can be lost during the simulation, a so-called re-initialization procedure needs to be performed. The LS function can be re-initialized either by performing a geometrical technique such as the fast marching method introduced by Sethian (1996), or solving an Eikonal equation introduced by Sussman et al. (1994). Although the LS method is



(a) Surface methods



(b) Volume methods

Figure 3.1 A schematic demonstration of the difference between the surface and volume methods for the numerical representation of an interface

very robust, but there is no guaranty for the area/volume conservation if the numerical method applied to solve the LSA equation (2.6) is not preciseness enough, see e.g. (Osher & Fedkiw 2003). The particle level set method (PLS) which was introduced by Enright et al. (2002), is a different way for keeping the accuracy of the LS method. Each side of the interface in this method is assigned a distinguished set of the massless particles which can be advected through the domain in a lagrangian way. As the particles preserve the material characteristics in time, they can be used to reconstruct the interface in the regions where an area/volume loss (or gain) occurs.

The marker and cell (MAC) method introduced by Harlow & Welch (1965), as well as the volume of fluid (VOF) method introduced by Hirt & Nichols (1981), are the common methods which can be classified under the category of the volume methods. The MAC method represents the phases on the either sides of the interface by a set of the massless particles which are advected through the domain in a Lagrangian way.

The interface is then reconstructed in the multi-phase cells using the distribution density of the particles. This method is computationally very expensive due to requiring a large number of the particles. Moreover, needing to add additional particles for making an accurate simulation of the interface stretch, is an issue which reduces the robustness of the method. In the VOF method, an indicator function is assigned to the phases at the either sides of the interface. The indicator function is commonly the volume fraction or the mass fraction of one of the phases. Therefore, it has a Heaviside distribution over the domain. The interface kinematics is simulated in this method by solving an advection equation for the indicator function. In the case of using a lower-order spatial discretization method, the Heaviside distribution of the indicator function is numerically smeared out. The region over which, this incorrect interface diffusion takes place, can be even developed by the velocity gradient in the direction normal to the interface. On the other hand, a higher-order numerical representation of the Heaviside distribution can lead to a numerical instability. Therefore, a lower-order method is used accompanying to applying an interface reconstruction technique in order to prevent the development of the diffusion region. The common interface reconstruction techniques include reconstructing an interface of degree zero introduced by Hirt & Nichols (1981), and reconstructing an interface of degree 1 introduced by Youngs (1982). Another common technique introduced by Ubbink (1997), is to compress the interface by adding a compression term to the advection equation. Although the implementation of the VOF method is rather straightforward, the interface reconstruction techniques always reduce the accuracy of the curvature calculation. But the main advantage of this method is to satisfy the area/volume conservation. Sussman & Puckett (2000) used this property of the VOF method, together with the ability of the LS method in a precise calculation of the curvature, for developing the idea of the coupled LS and VOF method (CLSVOF). The interface kinematics in this method is simulated by solving the VOF advection equation. The normal vector which is used for a piecewise linear interface reconstruction, is obtained using the LS function. The updated interface is then used for a geometrical re-initialization of the LS function. Comparing the mentioned methods in terms of the accuracy, robustness and ease of the implementation, one can conclude that the classical LS method is an appropriate choice when there is a possibility of solving the LSA equation (2.6). Accordingly, the LS method is used in the present research, as a higher-order DG method is applied for solving the governing equations.

3.2 Signed Distance Property of the Level Set Functions

As it was mentioned in section 2.1 SDLS functions are used for constructing the distributions for which, the distance to the interface is a parameter. For instance, the smoothed Heaviside functions such as (4.5.1) use the SDLS functions in order to keep the smoothness within a region of a certain width ϵ , see (Osher & Fedkiw 2003).

$$\mathcal{H}_\epsilon(\phi) = \begin{cases} 0, & \phi < -\epsilon, \\ \frac{1}{2} + \frac{\phi}{2\epsilon} + \frac{1}{2\pi} \sin\left(\frac{\pi\phi}{\epsilon}\right), & -\epsilon \leq \phi \leq \epsilon \\ 1, & \phi > \epsilon, \end{cases} \quad (3.1)$$

The reason of approximating the Heaviside function by the smoothed Heaviside functions is that the numerical representation of the exact Heaviside function may produce spurious spatial oscillations. These oscillations which are commonly referred as the Gibbs phenomenon, become even worth in the case of using higher-order numerical methods.

An SDLS function keeps its signed distance property if and only if the advection field satisfies the following condition,

$$\nabla u_n \cdot \nabla \phi = 0, \quad (3.2)$$

where u_n is the component of the velocity field in the direction normal to the LS function, obtained as,

$$u_n = \mathbf{u} \cdot \nabla \phi. \quad (3.3)$$

According to this condition, an SDLS function remains signed distance if u_n does not have any spatial variation in the direction normal to the LS function, see (Raessi et al. 2007).

If the advection field does not satisfy the condition (3.2), the signed distance property can be recovered by performing a re-initialization procedure that of course must not move the interface. It means that the re-initialization procedure is supposed to affect the LS function except its zero iso-value. The re-initialization can be mathematically modeled by an equation, namely the re-initialization equation. In order to present an instructive derivation of this equation, one can start with considering the following hyperbolic equation which describes the motion of the iso-values of an LS function in their normal directions,

$$\frac{\partial \phi}{\partial t} + (u_n \mathbf{N}) \cdot \nabla \phi = 0, \quad (3.4)$$

where \mathbf{N} represents the normal vector to each of the iso-values of the LS function, see e.g. (Osher & Fedkiw 2003). Since,

$$\mathbf{N} \cdot \nabla \phi = \frac{\nabla \phi}{|\nabla \phi|} \cdot \nabla \phi = \frac{|\nabla \phi|^2}{|\nabla \phi|} = |\nabla \phi|, \quad (3.5)$$

equation (3.4) can be rewritten as,

$$\frac{\partial \phi}{\partial t} + u_n |\nabla \phi| = 0, \quad (3.6)$$

By solving the equation (3.6) equation in a time interval Δt , the local value of ϕ increases by $(\Delta t)u_n$ times the value of its local gradient. In order to solve an Eikonal equation of the form $|\nabla \phi| = 1$, one can follow a pseudo-time stepping approach and

solve an equation of the form of equation (3.6) with $u_n = 1$ and an additional source term 1 as,

$$\frac{\partial \phi}{\partial \tau} + |\nabla \phi| = 1, \quad (3.7)$$

where τ is a pseudo-time, see (Rouy & Tourin 1992). By solving equation (3.7) in a pseudo-time interval $\Delta\tau$, the local value of ϕ increases by the difference of the value of its local gradient and 1. It should be noted that the LS function has the negative sign on the opposite side of the interface. Therefore, equation (3.7) takes the following form in the region $\phi < 0$,

$$\frac{\partial \phi}{\partial \tau} - |\nabla \phi| = -1, \quad (3.8)$$

Equations (3.7) and (3.8) together with the condition that the interface should not be affected by the re-initialization, can be combined into the following compact form proposed by Sussman et al. (1998),

$$\frac{\partial \phi}{\partial \tau} + \text{Sign}(\phi^0)(|\nabla \phi| - 1) = 0, \quad (3.9)$$

$$\phi(\mathbf{x}, 0) = \phi^0, \quad (3.10)$$

where $\text{Sign}(\phi^0)$ is a Signum function which is defined as,

$$\text{Sign}(\phi^0) = \begin{cases} -1 & \phi^0 < 0, \\ 0 & \phi^0 = 0, \\ 1 & \phi^0 > 0 \end{cases} \quad (3.11)$$

Sussman et al. (1999) rewrote the LSRI equation into the following more illuminating form,

$$\frac{\partial \phi}{\partial \tau} + \mathbf{w} \cdot \nabla \phi = \text{Sign}(\phi^0), \quad (3.12)$$

where \mathbf{w} is the characteristic velocity of the hyperbolic equation (3.12) and defined as,

$$\mathbf{w} = \text{Sign}(\phi^0) \frac{\nabla \phi}{|\nabla \phi|}. \quad (3.13)$$

Inclusion of the Signum function in the definition of the characteristic velocity implies that the vector \mathbf{w} points always outward the interface either within the region of $\phi^0 < 0$ or within the region of $\phi^0 > 0$. It means that the re-initialization of the LS function is started from the interface.

As it was mentioned before in the present section, the numerical representation of a jump may produce spurious spatial oscillations. Accordingly, for solving the LSRI equation (3.9), a smoothed Signum function as an approximate to the exact Signum function (3.11) is used. Although the following infinitely smoothed Signum function is commonly used in the literature, see e.g. Sussman et al. (1994),

$$\text{Sign}_{\epsilon \rightarrow \infty}(\phi^0) = \frac{\phi^0}{\sqrt{(\phi^0)^2 + \epsilon^2}} \quad (3.14)$$

but we used the following finitely smoothed Signum function in the present research in order to directly adjust the smoothing width,

$$Sign_{\epsilon}(\phi^0) = \begin{cases} -1 & \phi^0 < -\epsilon, \\ \frac{\phi}{\epsilon} + \frac{1}{\pi} \sin\left(\frac{\pi\phi^0}{\epsilon}\right) & -\epsilon \leq \phi^0 \leq \epsilon, \\ 1 & \epsilon < \phi^0, \end{cases} \quad (3.15)$$

which is resulted from the formulation (4.5.1). If the slope of the LS function is less or much less than 1, the smoothing width of the smoothed Signum function increases and consequently the speed of the characteristic lines is reduced. But if the slope is much higher than 1, the smoothed Signum function becomes too steep that may produce spurious spatial oscillations. In order to overcome this problem, Peng et al. (1999) proposed the following infinitely smoothed formulation in terms of the updated LS function ϕ instead of the initial LS function ϕ^0 ,

$$Sign_{\epsilon \rightarrow \infty}(\phi) = \frac{\phi}{\sqrt{(\phi)^2 + (|\nabla\phi|\epsilon)^2}}. \quad (3.16)$$

Multiplying ϵ by $|\nabla\phi|$ in the formulation (3.16), modifies the smoothing width in order to prevent the spurious spatial oscillations. As it was mentioned before, in the present research we prefer to use a smoothed Signum function with a finite width. Accordingly, the following formulation is constructed,

$$Sign_{\epsilon}(\phi) = \begin{cases} -1 & \phi < -|\nabla\phi|\epsilon, \\ \frac{\phi}{|\nabla\phi|\epsilon} + \frac{1}{\pi} \sin\left(\frac{\pi\phi}{|\nabla\phi|\epsilon}\right) & -\epsilon \leq \phi \leq |\nabla\phi|\epsilon, \\ 1 & |\nabla\phi|\epsilon < \phi \end{cases} \quad (3.17)$$

3.3 Higher-Order Numerical Schemes for Solving the Governing Equations

The governing equations of the present research can be summarized as follows,

- Continuity equation (2.27):

$$\nabla \cdot \mathbf{u} = 0$$

- Momentum equation (2.28):

$$\frac{\partial \mathbf{u}}{\partial t} + \nabla \cdot (\mathbf{u}\mathbf{u}) = -\frac{\nabla P}{\rho(\phi)} + \frac{1}{Re} \frac{\nabla \cdot (2\mu(\phi)\mathbf{D})}{\rho(\phi)} + \frac{\sigma\kappa\mathbf{n}\delta_{\mathcal{I}}}{\rho(\phi)We} + \frac{\mathbf{e}_g}{Fr^2},$$

- Level set advection equation (2.6):

$$\frac{\partial \phi}{\partial t} + \mathbf{u} \cdot \nabla \phi = 0,$$

- Level set re-initialization equation (3.9):

$$\begin{aligned}\frac{\partial \phi}{\partial \tau} + \text{Sign}(\phi^0)(|\nabla \phi| - 1) &= 0, \\ \phi(\mathbf{x}, 0) &= \phi^0,\end{aligned}$$

These governing equations consist of different types of the differential terms including temporal term, divergence term, diffusion term and source term, each of which, requires a certain consideration concerning the numerical solution of the equations. Therefore, the fundamental techniques used for solving all of these equations can be described by considering the numerical solution to a general scalar transport equation as,

$$\underbrace{\frac{\partial \rho \varphi}{\partial t}}_{\text{temporal term}} + \underbrace{\nabla \cdot (\rho \varphi \mathbf{u})}_{\text{advection term}} = \underbrace{\nabla \cdot (\Gamma \nabla \varphi)}_{\text{diffusion term}} + \underbrace{s_\varphi}_{\text{source term}}, \quad (3.18)$$

where φ denotes the unknown scalar, $\mathbf{u}(\mathbf{x}, t)$ denotes the velocity, ρ denotes the fluid density and Γ denotes the diffusion coefficient.

A numerical method for solving a scalar transport equation commonly consists of two consecutive stages, namely spatial discretization and temporal integration. The spatial discretization consists of the representation of the solution over a discrete domain and approximating the spatial differential terms in order to convert the PDE to a system of temporal ODEs. The solution in the next time step can be then obtained by performing a temporal integration.

3.3.1 Spatial Discretization Methods

The spatial discretization methods are principally classified into three categories including the FD method, the FV method and the family of the spectral methods. These methods are distinguished mainly based on the senses in which, they are consistent with the spatial differential terms. It should be noted that following (Karniadakis & Sherwin 2005), the FE method is also classified under the category of the spectral methods in the present dissertation. The starting point in the procedure of a spatial discretization is converting the continuous domain to a discrete domain, over which the numerical solution is represented and the spatial differential terms are approximated. As a discrete domain looks like a network of points, it is commonly termed the numerical grid. A discrete domain in the finite difference (FD) method, see e.g. (Ferziger & Peric 2002), and the pseudo-spectral method, see e.g. (Fornberg 1998), consists of a set of nodes. The FD method is designed for being implemented on the structured grids. A structured grid can have a curvilinear pattern, such as a polar grid around a circle in a circular domain. In this case in order to perform the FD computations, a curvilinear pattern is required to be mapped to its corresponding Cartesian pattern in a so-called computational domain. Therefore for instance, the polar grid within a circular domain with a circle at its center, is mapped to a cartesian grid within a rectangular domain. As a result, the circle at the center of the circular

physical domain is mapped to a line on the boundary of the rectangular computational domain, see e.g. (Thompson et al. 1999). This signifies that as mapping from physical domains with complex geometries to their corresponding computational domains are very difficult, the FD method is not appropriate for such cases. In the finite volume (FV) method, see e.g. (Ferziger & Peric 2002), and the spectral methods, see e.g. (Karniadakis & Sherwin 2005), a discrete domain is composed of a set of the sub-domains termed cells. There is basically no restriction on the geometry of the cells.

Why applying the higher-order schemes? Generally speaking, increasing the grid resolution is an essential way for improving the solution accuracy in all of the numerical methods. However, the rate of convergence is in a direct relation to the order of the spatial discretization. In the case of lower-order schemes, increasing the grid resolution leads to an error reduction that is relatively small compared to the additional computational effort, see e.g. (Shu 2003). An instructive interpretation to this behavior can be made in the context of the Fourier analysis, see e.g. (Ainsworth 2004). The higher-order spatial discretizations in principle give the ability of resolving the modes of the solution which have higher wave numbers within the same stencil. Therefore, increasing the grid resolution results in a error reduction with a higher rate. Another consequence of using the higher-order schemes is reducing the numerical dissipation and dispersion errors, see e.g. (Karniadakis & Sherwin 2005). These errors are quite determinant in making accurate solutions to the hyperbolic conservation laws. For instance, considering a one-dimensional (1D) wave equation for an arbitrary variable φ ,

$$\frac{\partial \varphi}{\partial t} + a \frac{\partial \varphi}{\partial x} = 0, \quad (3.19)$$

with a as the wave speed, the dissipation error reduces the amplitude of the wave leading to the dissipation of the wave (figure 3.2) and the dispersion error affects the speed of the wave and produces spurious oscillations (3.3). Therefore, if the solution to the equation (3.32) has high dissipation and dispersion errors, it results in simulating a dissipative wave moving with a wrong speed. Hence, a major advantage of a numerical method is providing a context, in which a higher-order spatial discretization can be formulated efficiently.

3.3.1.1 Finite Difference Method

In the FD method, the solution is represented by its values on a set of the nodes distributed over the physical domain of solution on a structured pattern, as shown in figure 3.4. Each value can be associated with a higher-order polynomial distribution within a certain stencil (a set of the neighbor nodes) over the domain. Therefore, higher convergence rates can be achieved by using higher-order FD methods. It can be shown that the higher-order FD methods yield quasi-exponential convergence rate for the solutions with limited wave numbers, see (Sarson 2007). The spatial differential terms are discretized using formulations based on either a Taylor series expansion or a polynomial fitting. The higher-order discretizations can be formulated

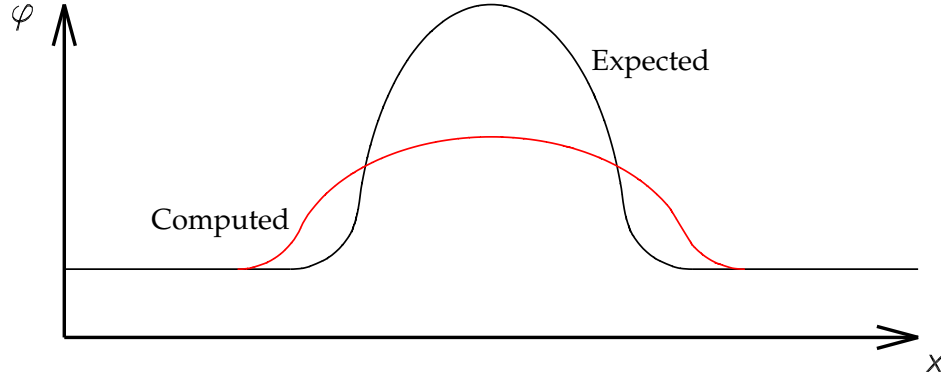


Figure 3.2 A schematic representation of the dissipation error for a 1D wave propagation

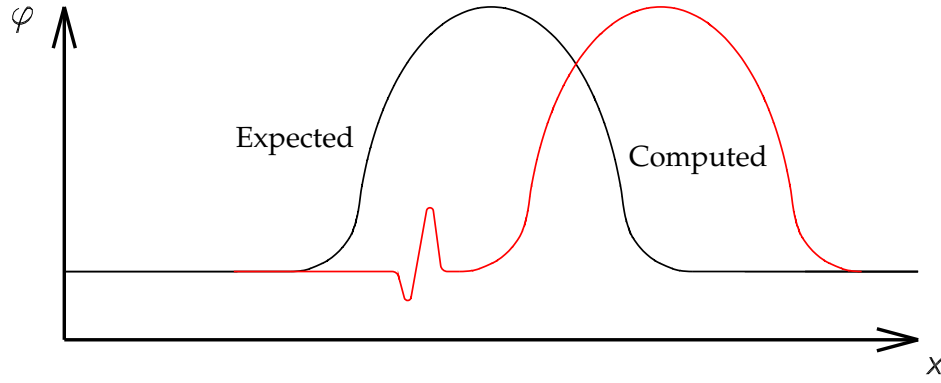


Figure 3.3 A schematic representation of the dispersion error for a 1D wave propagation

by involving more neighbor nodes. For instance, a sixth-order discretization of the first derivative can be written over an equidistance grid as,

$$\begin{aligned} \frac{\partial \varphi}{\partial x} \Big|_{i,j} &= \frac{-\varphi_{i-3,j} + 9\varphi_{i-2,j} - 45\varphi_{i-1,j} + 45\varphi_{i+1,j} - 9\varphi_{i+2,j} + \varphi_{i+3,j}}{60h} \\ &+ O(h^6), \end{aligned} \quad (3.20)$$

where i and j denote the node indices and $h = |x_{i,j} - x_{i-1,j}|$ denoted the uniform grid resolution. Such a wide stencil used in this formulation can be inappropriate from different aspects. For instance, this symmetric stencil has to be broken close to the boundary. Therefore, close to the boundary, one need to use either a lower-order approximation over a smaller symmetric stencil, or using a 6th-order approximation but over an asymmetric stencil. Several algorithms have been developed for generating compact formulations corresponding to the same order of approximation. The compact formulations are implicit as they additionally use the unknown values of

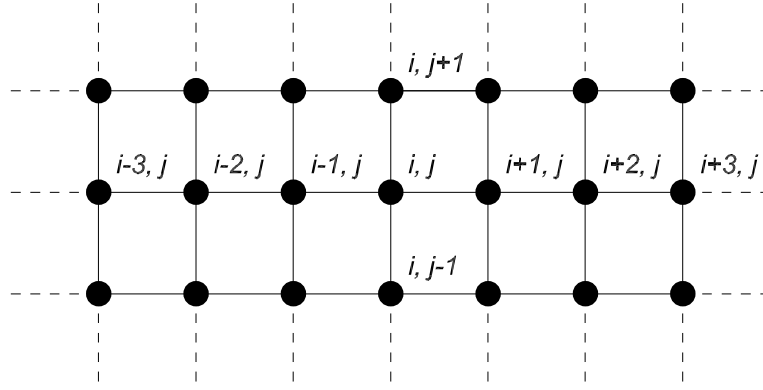


Figure 3.4 A 2D discrete domain used for an FD discretization

derivatives at neighbor nodes. For instance, the same order of approximation made by formulation (3.20), can be achieved using a compact formulation,

$$\frac{1}{3} \frac{\partial \varphi}{\partial x} \Big|_{i-1,j} + \frac{\partial \varphi}{\partial x} \Big|_{i,j} + \frac{1}{3} \frac{\partial \varphi}{\partial x} \Big|_{i+1,j} = \frac{\varphi_{i-2,j} - 28\varphi_{i-1,j} + 28\varphi_{i+1,j} - \varphi_{i+2,j}}{36h} + O(h^6). \quad (3.21)$$

The algorithms used for generating such formulations can be found in (Fornberg 1998). It is obvious that even the compact formulations use rather wide stencils.

3.3.1.2 Finite Volume Method

In the classical FV method, the solution is represented in each cell by a cell-averaged value which is assigned to the center of the cell. As this value maybe associated with uniform distribution over each cell, the solution representation in this method is of degree 1. Moreover, the solution is discontinuous across the borders of the cells (see figure 3.5). In the FV method, the main idea for the spatial discretization is to write

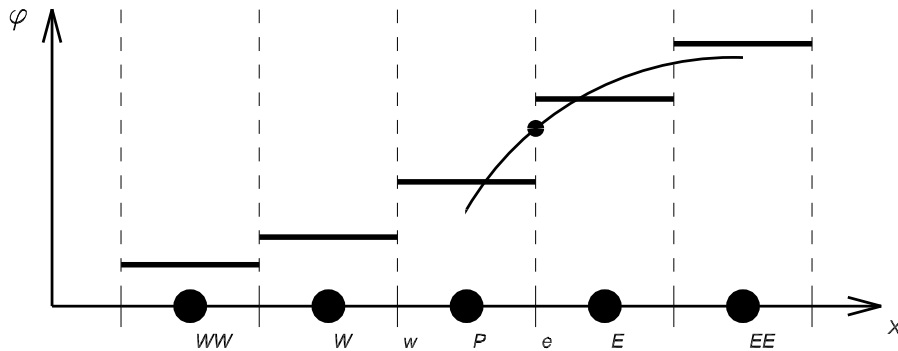


Figure 3.5 Representation of a solution on a 1D discrete domain using an FV method

the integral forms of the spatial differential terms in each cell and considering the integration over whole the domain as the summation of these integrals. This reflects

the local character of this method. For instance, following (Ferziger & Peric 2002), the integral form of equation (3.18) over a cell Ω_i can be written as,

$$\frac{d}{dt} \int_{\Omega_i} \rho \bar{\varphi} d\mathcal{V} + \int_{\partial\Omega_i} (\rho \bar{\varphi} \mathbf{u}) \cdot \mathbf{n}_s dS = \int_{\partial\Omega_i} (\Gamma \nabla \bar{\varphi}) \cdot \mathbf{n}_s dS + \int_{\Omega_i} s_{\bar{\varphi}} d\mathcal{V}, \quad (3.22)$$

where $\bar{\varphi}$ denotes the cell-averaged value of φ and \mathbf{n}_s denotes the normal vector to the cell border. The integrals can be evaluated numerically using a higher-order approximation. The neighbor cells in this method are connected to each other through the necessity of calculating a single flux function across each of their common borders. The explicit specification of the flux functions across the cell borders signifies the conservative property of the FV method. In order to calculate a single numerical flux across a cell border, a continuous solution is required to be reconstructed there off. In figure 3.5, a solution reconstruction at cell border (e) using the values assigned to points (P), (E) and (EE), is represented schematically. Different FV methods are mainly distinguished by the schemes they use for the solution reconstruction. The higher-order reconstruction schemes are often based on using a polynomial fitting, such as the 2^{nd} -order MUSCL scheme (Monotone Upwind Schemes for the Scalar Conservation Laws) proposed by (Leer 1979), the 3^{rd} -order QUICK scheme (Quadratic Upstream Interpolation for Convective Kinetics) proposed by (Leonard 1979), the 3^{rd} -order ENO scheme (Essentially Non-Oscillatory Scheme) proposed by (Harten et al. 1987) and the 3^{rd} -order WENO scheme (Weighted Essentially Non-Oscillatory Scheme) proposed by (Liu et al. 1996). The MUSCL, ENO and WENO schemes are specifically designed to handle steep variations in the solution such as shocks. Although the first version of the WENO scheme was 3^{rd} -order, several higher-order versions (up to 11^{th} -order, see (Balsara & Shu 2000)) were proposed afterwards. The first step in the ENO and WENO schemes is making a set of the polynomial fittings over a number of different stencils with the same sizes. For instance, a polynomial fitting of degree 3 at cell border (e), can be made using points {P, E, EE} or {W, P, E} or {WW, W, P}. The ENO scheme selects only one of these polynomials which makes smoothest solution, whereas the WENO scheme makes a convex combination of all of these polynomials resulting an optimal smoothness of the solution, see e.g. (Osher & Fedkiw 2003). In this way, the WENO scheme can make a smoother solution reconstruction. It should be noted that although several higher-order schemes are developed for reconstructing the solution at the cell borders, the solution representation over the domain is still zeroth-order. Therefore, although a considerable error reduction can be achieved by using the higher-order solution reconstruction schemes, the spatial convergence rate is almost one.

3.3.1.3 Spectral Methods

The central idea of the spectral methods is to consider the solution as a certain composition of a spectrum of prescribed analytical functions. Accordingly, this type of methods can be considered as mathematical spectralizers. In this sense, the term "spectralization" shall be used for the spectral representation of a solution. The main

motivation for designing the spectral methods is introducing more degrees of freedom to each cell in order to create an efficient higher-order spatial discretization. For instance, the solution $\varphi(\mathbf{x}, t)$ can be spectralized as,

$$\varphi(\mathbf{x}, t) \approx \tilde{\varphi}(\mathbf{x}, t) = \sum_{j=1}^{N_{DoF}} \hat{\varphi}_j(t) \vartheta_j(\mathbf{x}), \quad (3.23)$$

where $\vartheta_j(\mathbf{x})$ represents the analytical functions which are commonly termed as the basis functions, $\hat{\varphi}_j(t)$ represents the corresponding coefficients, and N_{DoF} denotes the number of the numerical degrees of freedom inside the domain, over which the solution is spectralized. As the basis functions are prescribed, the solution is obtained by computing the unknown coefficients $\varphi_j(t)$ following a procedure which is described in detail subsequently. It can be shown that the spectral methods yield exponential convergence rate, see e.g. (Karniadakis & Sherwin 2005).

There are three main characteristics, based on which the different spectral methods are distinguished:

- The first one is the domain, over which the solution is spectralized. In the spectral element methods such as the DG method, the spectralization is performed over each cell separately. Whereas in the other methods such as the FE method and the Fourier spectral (FS) method, the spectralization is made globally or over a wider stencil.
- The second characteristics is the type of the basis functions and the domain within which, they are defined. In some of the of the spectral methods such as the FE and the DG methods, the basis functions are defined (locally supported) within each cell. Whereas in the other methods they are defined within wider stencils, such as some types of the FE method, or globally over the whole domain, such as the FS method. As stated in (Fornberg 1998), the most suitable basis functions for the periodic problems are the trigonometric functions. On the other hand, the orthogonal polynomials are proven to be appropriate for the non-periodic problems. A set of the non-orthogonal polynomials can be orthogonalized applying the Gram-Schmidt algorithm which is described subsequently.
- The third issue is the technique applied for determining the unknown coefficients $\hat{\varphi}_n(t)$. For the general non-periodic problems, the unknown coefficients are determined applying the method of weighted residuals which is explained subsequently.

3.3.1.3.1 Method of Weighted Residuals

As the approximate solution $\tilde{\varphi}(\mathbf{x}, t)$ does not necessarily satisfy equation (3.18), its substitution into the equation results in the appearance of a residual term $\mathcal{R}(\tilde{\varphi})$,

$$\frac{\partial \rho \tilde{\varphi}}{\partial t} + \nabla \cdot (\rho \tilde{\varphi} \mathbf{u}) - \nabla \cdot (\Gamma \nabla \tilde{\varphi}) - s_{\tilde{\varphi}} = \mathcal{R}(\tilde{\varphi}). \quad (3.24)$$

The main idea of the method of weighted residuals is looking for an approximate solution that satisfies a restriction imposed on the residual function. The restriction is placed by equating the Legendre inner product of the residual function and a test (weight) function, to zero, as,

$$\langle \chi_j(\mathbf{x}), \mathcal{R}(\mathbf{x}) \rangle = \int_{\Omega} \chi_j(\mathbf{x}) \mathcal{R}(\mathbf{x}) d\mathcal{V} = 0, \quad j = 1, \dots, N_{DoF}. \quad (3.25)$$

Then, equation (3.24) takes a form as,

$$\int_{\Omega} \chi_j(\mathbf{x}) \left[\frac{\partial \rho \tilde{\varphi}}{\partial t} + \nabla \cdot (\rho \tilde{\varphi} \mathbf{u}) - \nabla \cdot (\Gamma \nabla \tilde{\varphi}) - s_{\tilde{\varphi}} \right] d\mathbf{x} = \int_{\Omega} \chi_j(\mathbf{x}) \mathcal{R}(\tilde{\varphi}) d\mathcal{V}, \quad (3.26)$$

$$j = 1, \dots, N_{DoF},$$

which is in fact a set of N_{DoF} equations. There are different methods of weighted residuals which are distinguished based on the type of the test function they employ, see (Karniadakis & Sherwin 2005). The common ones are listed in the table 3.1.

Table 3.1

Methods of weighted residuals

Type of the Method	Test Function
Finite Volume (Sub-domain)	$\chi_j(\mathbf{x}) = \begin{cases} 1, & \text{inside } \Omega_k^h \\ 0, & \text{outside } \Omega_k^h \end{cases}$
Least-Squares	$\chi_j(\mathbf{x}) = \frac{\partial \mathcal{R}}{\partial \hat{\varphi}_j}$
Collocation	$\chi_j(\mathbf{x}) = \delta(\mathbf{x} - \mathbf{x}_j)$
Galerkin	$\chi_j(\mathbf{x}) = \vartheta_j(\mathbf{x})$
Petrov-Galerkin	$\chi_j(\mathbf{x}) = \theta_j(\mathbf{x}) (\neq \vartheta_j(\mathbf{x}))$

Finite Volume (Sub-domain) Method The FV method can be principally considered as a method of weighted residuals with the simple test function defined in the table 3.1, where $\Omega_{h,k}$ denotes each of the cells.

Least-square Method In this method the coefficients $\hat{\varphi}$ are determined based on the minimization of the L^2 -norm of the residual function. It is obvious that the minimization of $\langle \mathcal{R}, \mathcal{R} \rangle$ is principally equivalent to equating $\langle \chi_j = \partial \mathcal{R} / \partial \hat{\varphi}, \chi_j = \partial \mathcal{R} / \partial \hat{\varphi} \rangle$ to zero. Therefore, the least-square method takes the form of a method of weighted residuals by using $\partial \mathcal{R} / \partial \hat{\varphi}$ instead of the residual function \mathcal{R} itself.

Collocation Method In this method which is also referred to as the nodal method, the unknown coefficients $\tilde{\varphi}$ are obtained by requiring the residual function to be zero at a set of nodes distributed over the domain. As the unknown coefficients in this

method are not determined at every point inside the domain, this method is also termed as the pseudo-spectral method.

Galerkin Method This method is distinguished by using the same formulation for the test and the basis functions. The FE method as well as the DG method are the prevalent spectral methods which are characterized by employing the Galerkin method of weighted residuals.

3.3.1.4 Numerical integration

In the procedures of applying the spatial discretization methods as well as in the post-processing stages, one may require the numerical evaluation of definite integrals. The numerical integration methods are commonly referred in the literature as the quadrature rules. Among the different proposed methods, the Gaussian quadrature rules are highly efficient because of the minimal number of evaluations. In principle, a Gaussian quadrature rule consists of evaluating the integrand at a certain set of the integral points (quadrature points) and making a weighted summation of the calculated values, see e.g. (Kress 1998). For instance, a 1D Gaussian quadrature rule over a domain $[-1,1]$ can be written as,

$$\int_{-1}^1 f(x)dx \approx \sum_{i=1}^n \omega_i f(x_i), \quad (3.27)$$

where ω_i represents the prescribed weighting functions and x_i represents the quadrature points which are distributed in a certain pattern within the domain of integration. The 1D version of a Gaussian quadrature rule can be simply extended to the corresponding multi-dimensional version by making the same distribution of the quadrature points over a multi-dimensional domain, and employing a multi-dimensional weighting function.

If the integrand has a Heaviside distribution, the accuracy of the integration can not be improved by increasing the order of quadrature rule. In this case, the accuracy can be improved by performing a multistage division of each cell and applying a lower-order quadrature rule over each of the sub-cells. This technique is commonly termed as the Brute Force integration.

3.3.2 Temporal Discretization Methods

As it was mentioned before, the spatial discretization of an unsteady PDE, converts it to a system of the 1st-order temporal ODEs which is required to be converted to a system of the algebraic equations by applying a temporal discretization method. Considering the ODE,

$$\varphi_t \equiv \frac{d\varphi}{dt} = f(\varphi, t), \quad (3.28)$$

the aim is constructing a sequence of values $\tilde{\varphi}^0, \tilde{\varphi}^1, \dots, \tilde{\varphi}^n$ such that,

$$\begin{aligned}\tilde{\varphi}^n &\approx \varphi(t_n), \\ f^n &= f(\tilde{\varphi}^n, t).\end{aligned}\tag{3.29}$$

There are two major families of the discretization methods applied for solving ODEs, i.e. the linear multi-step (LMS) methods and the Runge-Kutta (RK) methods, see (Trefethen 1996).

3.3.2.1 Linear Multi-Step Methods

In this type of methods, each new value $\tilde{\varphi}^{n+1}$ is calculated using the values of a number of the other time steps. The method is explicit if it uses only the values of the previous time steps, and is implicit if uses the values of both the previous and later time steps. A general form of the LMS methods can be written as,

$$\sum_{j=0}^s \alpha_j \tilde{\varphi}^{n+j} = \Delta t \sum_{j=0}^s \beta_j f^{n+j}\tag{3.30}$$

where s denotes the number of steps and α_j and β_j are constants. In addition, $\alpha_s = 1$ and either $\alpha_0 \neq 0$ or $\beta_0 \neq 0$. If $\beta_s = 0$ the method is explicit, such as the Adams-Bashforth methods. Otherwise, it is implicit, such as the Adams-Moulton methods. Although the implementation of the implicit methods are more difficult than the explicit ones, they are much more stable. The main advantage of the this type of the methods is that they have only one function evaluation in each time step. A disadvantage of the higher-order methods of this type, is the problem of initialization. For instance, in order to march from $\tilde{\varphi}^0$ to $\tilde{\varphi}^1$, one may require the value of $\tilde{\varphi}^{-1}$ which is not available. This of course can be resolved by using a 1st-order method for the first time step.

3.3.2.2 Runge-Kutta Methods

These methods are the one-step multi-stage schemes, where the function f may be evaluated at any number of the stages which marching from $\tilde{\varphi}^n$ to $\tilde{\varphi}^{n+1}$. This multi-stage evaluation makes the RK methods much more stable than the LMS methods. A general form of the explicit RK method can be written as,

$$\begin{aligned}
\tilde{\varphi}^{n+1} &= \tilde{\varphi}^n + \sum_{i=1}^s b_i k_i, \\
k_1 &= (\Delta t) f(\tilde{\varphi}^n, t_n), \\
k_2 &= (\Delta t) f(\tilde{\varphi}^n + a_{21} k_1, t_n + c_2 \Delta t), \\
&\vdots \\
k_s &= (\Delta t) f(\tilde{\varphi}^n + a_{s,s-1} k_{s-1}, t_n + c_s \Delta t),
\end{aligned} \tag{3.31}$$

where a_{ij} forms the RK matrix, b_i forms the vector of weights and c_i forms the vector of nodes. The RK methods can be represented compactly using Butcher tableau as,

0				
c_2	a_{21}			
c_3	a_{31}	a_{32}		
\vdots	\vdots	\vdots	\ddots	
c_s	a_{s1}	a_{s2}	\cdots	$a_{s,s-1}$
	b_1	b_2	\cdots	$b_{s-1} \quad b_s$

Although the RK methods were originally designed as an explicit method, there is also an implicit version, which can be represented as,

c_1	a_{11}	a_{12}	\cdots	$a_{1,s}$
c_2	a_{21}	a_{22}	\cdots	$a_{2,s}$
\vdots	\vdots	\vdots	\ddots	\vdots
c_s	a_{s1}	a_{s2}	\cdots	a_{ss}
	b_1	b_2	\cdots	b_s

Generally, a possible consequence of using the higher-order schemes is the occurrence of spurious numerical oscillation. Among the various types of the RK methods, the total variation diminishing (TVD) RK method can conditionally prevent such undesirable effects, see (Gottlieb & Shu 1998). The word "Conditionally" was used, as the the TVD property of a temporal discretization method should be verified in conjunction with the spatial discretization method.

Considering a one-dimensional wave equation

$$\varphi_t + a\varphi_x = 0,$$

the total variation of the variable φ can be calculated by,

$$TV = \int \left| \frac{\partial \varphi}{\partial x} \right| dx. \tag{3.32}$$

The temporal discretization of such an equation, in conjunction with the spatial discretization, is TVD if

$$TV(\varphi^{n+1}) \leq TV(\varphi^n). \quad (3.33)$$

It can be proven that a TVD method is monotonicity preserving and also a monotone numerical scheme is TVD (see (Harten 1983)). A numerical scheme is monotonicity preserving if the following properties are maintained in time (see e.g. (Hirsch 2007)),

- No creation of any new extremum in the spatial distribution of the solution
- No decreasing values of the local minimums and no increasing values of the local maximums

It can be shown that the 1st-order forward (Euler) temporal discretization method in conjunction with a proper spatial discretization method, is a basic TVD RK method, see (Gottlieb & Shu 1998). The Euler method can be written as,

$$\tilde{\varphi}_t = \frac{\tilde{\varphi}^{n+1} - \tilde{\varphi}^n}{\Delta t}. \quad (3.34)$$

Therefore, each stage of a multi-stage higher-order TVD RK method can be constructed by applying an Euler method and combining the results with the initial data using a convex combination (see (Osher & Fedkiw 2003)). The convex combination is a linear combination with the positive coefficients, where the summation of the coefficients is equal to one. For instance, the 3rd-order TVD RK method which is applied for the present study, can be constructed as (see (Gottlieb & Shu 1998)),

$$\begin{aligned} \tilde{\varphi}^1 &= \tilde{\varphi}^n + \Delta t f^1 \\ \tilde{\varphi}^2 &= \frac{3}{4}\tilde{\varphi}^n + \frac{1}{4}\tilde{\varphi}^1 + \frac{1}{4}\Delta t f^1 \\ \tilde{\varphi}^{n+1} &= \frac{1}{3}\tilde{\varphi}^n + \frac{2}{3}\tilde{\varphi}^2 + \frac{2}{3}\Delta t f^2 \end{aligned} \quad (3.35)$$

3.4 Discontinuous Galerkin Method

3.4.1 Solution Representation

As the spatial discretizations in the present research are made applying the DG method, this section is specifically assigned to explaining the corresponding concepts in more detail. The idea of the DG method was introduced for the first time in (Reed & Hill 1973) for solving the neutron transport equation. A major development in this method was performed by Cockburn and his coworkers in the context of solving the conservation laws, published as a series of papers concluded by (Cockburn & Shu 1998). The DG method can be classified in the category of spectral element methods. It means that the spectral representation of the solution in this method is made in a cell-wise manner. As a result, the solution is allowed to be discontinuous across the borders of the cells. For instance, figure 3.6 shows a DG-based representation of

a solution over a two-dimensional (2D) domain. Observing the picture, one can easily recognize the solution discontinuities. The amplitude of the discontinuities are adjusted by the sizes of the cells as well as the order of the spectral representation. The procedure of applying the DG method starts with approximating the physical

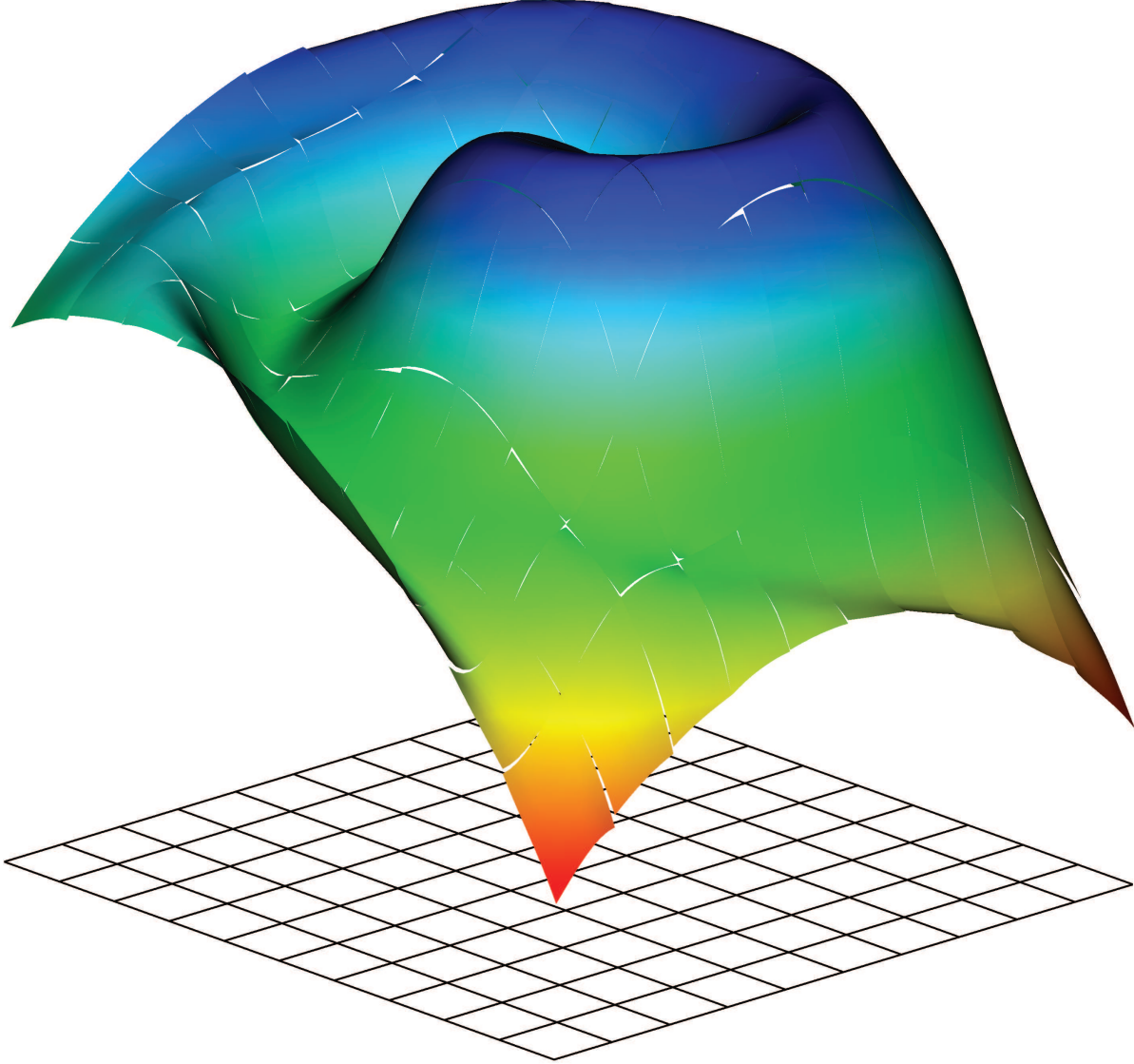


Figure 3.6 Example representation of a solution using a DG method with observed cell-boundary discontinuities

domain Ω bounded by $\partial\Omega$ by a discrete domain Ω_h bounded by $\partial\Omega_h$, consisting of N_C non-overlapping boundary conforming cells Ω_k^h . The DG field $\varphi_{hp,k}(\mathbf{x}, t)$ which is defined as the DG-based representation of a solution $\varphi(\mathbf{x}, t)$, can be constructed over each cell Ω_k^h employing an orthogonal basis polynomials space $\vartheta_j(\mathbf{x})$ as,

$$\varphi(\mathbf{x}, t)|_{\Omega_k^h} \approx \varphi_{hp}|_{\Omega_k^h}(\mathbf{x}, t) = \sum_{j=1}^{N_P} \hat{\varphi}(t)_{jk} \vartheta_{jk}(\mathbf{x}), \quad k = 1, \dots, N_C, \quad (3.36)$$

where N_P denotes the dimension of the orthogonal basis polynomials space. The letter h in the subscript hp , indicates the finite spatial resolution, and the letter p signifies the spectral representation. The dimension of the orthogonal basis polynomials space which is required for constructing a DG field of degree p , can be calculated as,

$$N_P = \frac{1}{D!} \prod_{1 \leq l \leq D} (p + l), \quad (3.37)$$

where D denotes the spatial dimension. In order to obtain the coefficients $\hat{\varphi}(t)$ corresponding the DG-based representation of a solution $\varphi(\mathbf{x}, t)$, the solution needs to be projected over the orthogonal basis polynomials space $\vartheta_{jk}(\mathbf{x})$ by performing an inner product as,

$$\begin{aligned} \hat{\varphi}(t)_{jk} &= \langle \varphi(\mathbf{x}, t), \vartheta_{jk}(\mathbf{x}) \rangle = \int_{\Omega_k^h} \varphi(\mathbf{x}, t) \vartheta_{jk}(\mathbf{x}) d\mathcal{V}, \\ j &= 1, \dots, N_P, \\ k &= 1, \dots, N_C, \end{aligned} \quad (3.38)$$

The procedure of solving an equation in the modal DG method is finalized by obtaining the coefficients $\hat{\varphi}(t)$. Then the solution values can be calculated at any arbitrary point within the domain in the postprocessing stage. This type of the DG method is used in the present research. There is also another alternative which is known as the nodal DG-based representation of the solution (see e.g. (Hesthaven 2008)). The basis functions employed in the this method, are a set of the Lagrange interpolation functions. In nodal DG method, the procedure of solving an equation is finalized by obtaining the solution values at the certain points within the cells.

3.4.1.1 Orthonormal Basis Polynomial Space

It is more convenient to normalize the OBPS by the \mathbb{L}^2 norm of each of the polynomials, resulting an orthonormal basis polynomial space. The OBPS used for the present research is constructed over a set of the monomials applying the Gram-Schmidt algorithm. A monomial can be defined as a polynomial with only one term. For instance, a set of the 2D monomials can be tabulated as,

$$\mathcal{P} = \left\{ \begin{array}{ccccccccc} & & & & & & & & 1, \\ & & & & & & & x, & y, \\ & & & x^2, & & xy, & & y^2, & \\ & x^3, & & x^2y, & & xy^2, & & y^3, & \\ x^4, & & x^3y, & & x^2y^2, & & xy^3, & y^4, & \\ x^5, & x^4y, & x^3y^2, & x^2y^3, & xy^4, & \dots & & & \end{array} \right\}$$

The GramSchmidt algorithm begins by selecting one of the polynomials, let say \mathcal{P}_1 , and finding a component of the second polynomial \mathcal{P}_2 which is orthogonal to the

first polynomial. This is performed by projecting the first polynomial on the second polynomial and subtracting this projection from the second polynomial as,

$$\begin{aligned}\vartheta_{1k} &= \frac{\mathcal{P}_1}{\|\mathcal{P}_1\|} \\ \theta_{2k} &= \mathcal{P}_2 - \langle \mathcal{P}_2, \vartheta_{1k} \rangle \vartheta_{1k} \\ \vartheta_{2k} &= \frac{\theta_{2k}}{\|\theta_{2k}\|}\end{aligned}$$

where ϑ_{2k} is orthonormal to ϑ_{1k} . In the same way, the j^{th} polynomial \mathcal{P}_j can become orthonormal to each of the $j - 1$ polynomials which are already orthonormal, as,

$$\begin{aligned}\theta_{jk} &= \mathcal{P}_j - \sum_{i=1}^{j-1} \langle \mathcal{P}_j, \vartheta_{ik} \rangle \vartheta_{ik} \\ \vartheta_{jk} &= \frac{\theta_{jk}}{\|\theta_{jk}\|}\end{aligned} \tag{3.39}$$

For instance, the polynomial space $\mathcal{P} := \{1, x, y, x^2, y^2, xy\}$ can be converted to an orthonormal polynomial space \mathbb{P} as (see (Emamy 2010)),

$$\Theta := \left\{ \frac{1}{2}, \frac{\sqrt{3}}{2}x, \frac{\sqrt{3}}{2}y, \sqrt{5} \left(\frac{3}{4}x^2 - \frac{1}{4} \right), \frac{3}{2}xy, \sqrt{5} \left(\frac{3}{4}y^2 - \frac{1}{4} \right) \right\}$$

3.4.1.2 Solution Discontinuity

As a result of the discontinuities of the solution across the borders of the cells, every point located on each border corresponds to a pair of the asymptotic values, namely the inner-cell value and the outer-cell value, denoted by $\varphi_{hp}^-(\mathbf{x}^k)$ and $\varphi_{hp}^+(\mathbf{x}^k)$, respectively. A conceptual representation of these values is given in figure 3.7 for a 1D problem. As it is indicated in this figure, the solution discontinuity is quantified by the jump operator $[[\varphi_{hp}]]_{x^k}$ which is defined as the difference between the inner- and the outer-cell values. The inner- and the outer-cell values are defined on the border $\partial\Omega_{h,k}$ of a cell $\Omega_{h,k}$ as,

$$\varphi_{hp}^-(\mathbf{x}^k) := \lim_{\boldsymbol{\xi}^k \rightarrow \mathbf{x}^k} \varphi_{hp}(\boldsymbol{\xi}^k), \quad \mathbf{x}^k \in (\Omega_k^h \setminus \partial\Omega_k^h), \tag{3.40}$$

$$\varphi_{hp}^+(\mathbf{x}^k) := \lim_{\boldsymbol{\xi}^k \rightarrow \mathbf{x}^k} \varphi_{hp}(\boldsymbol{\xi}^k), \quad \mathbf{x}^k \notin \bar{\Omega}_k^h, \tag{3.41}$$

where

$$\mathbf{x}^k \in (\Omega_k^h \setminus \partial\Omega_k^h) := \{\mathbf{x}^k \in \Omega_k^h : \mathbf{x}^k \notin \partial\Omega_k^h\},$$

and

$$\mathbf{x}^k \notin \bar{\Omega}_k^h \equiv \mathbf{x}^k \notin \left(\Omega_k^h \cup \partial\Omega_k^h \right) := \{\mathbf{x}^k : \mathbf{x}^k \in \Omega_k^h \vee \mathbf{x}^k \in \partial\Omega_k^h\}.$$

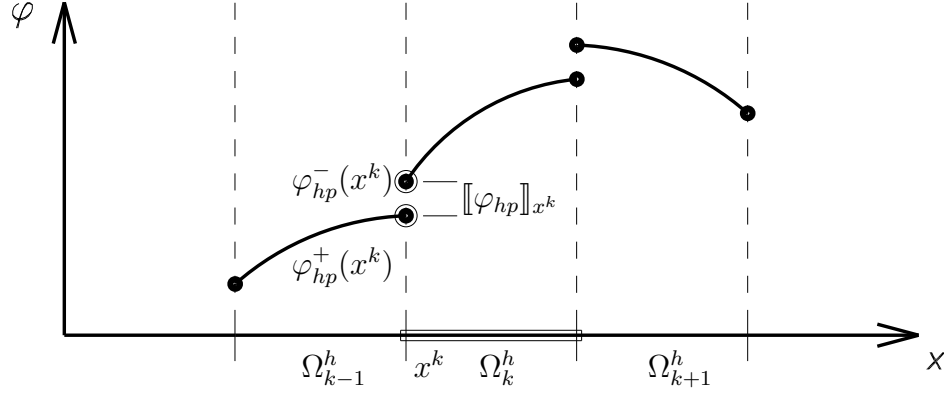


Figure 3.7 A schematic representation of the discontinuity of the solution across each of the cell borders

3.4.2 Spatial Discretization

The spatial discretization in the DG method is made by applying a Galerkin weighted residual method. For instance, an equation with the type (3.18) can be discretized by multiplying the N_P members of the OBPS ϑ_j to the equation 3.18 as,

$$\int_{\Omega_k^h} \vartheta_{jk}(\mathbf{x}) \left[\frac{\partial \rho \varphi_{hp}}{\partial t} + \nabla \cdot (\rho \varphi_{hp} \mathbf{u}_{hp}) - \nabla \cdot (\Gamma \nabla \varphi_{hp}) - s_\varphi \right] d\mathcal{V} = 0, \quad (3.42)$$

$$j = 1, \dots, N_P,$$

$$k = 1, \dots, N_C,$$

leading to a matrix equation as,

$$\underbrace{\rho \int_{\Omega_k^h} \vartheta_{jk}(\mathbf{x}) \left(\frac{\partial \varphi_{hp}}{\partial t} \right) d\mathcal{V}}_{\text{TM}} + \underbrace{\rho \int_{\Omega_k^h} \vartheta_{jk}(\mathbf{x}) (\nabla \cdot \underbrace{(\varphi_{hp} \mathbf{u}_{hp})}_{\text{Non-Linear Flux} := \mathbf{f}}) d\mathcal{V}}_{\text{AM}} - \underbrace{\Gamma \int_{\Omega_k^h} \vartheta_{jk}(\mathbf{x}) (\nabla \cdot (\nabla \varphi_{hp})) d\mathcal{V}}_{\text{DM}} - \underbrace{\int_{\Omega_k^h} \vartheta_{jk}(\mathbf{x}) (s_\varphi) d\mathcal{V}}_{\text{SM}} = 0, \quad (3.43)$$

where ρ and Γ are assumed to be constant. The matrices are of the size $1 \times N_P$. It is more convenient to consider the matrix corresponding to each of the terms separately as **TM**, **AM**, **DM** and **SM**, denoting the temporal matrix, the advection matrix, the diffusive matrix and the source matrix respectively.

Temporal Matrix (TM) Substituting the DG-based representation of φ into this tensor yields,

$$\begin{aligned}
\mathbf{TM} &:= \int_{\Omega_k^h} \vartheta_{jk}(\mathbf{x}) \left(\frac{\partial}{\partial t} \left(\sum_{i=1}^{N_P} \hat{\varphi}_{ik}(t) \vartheta_{ik}(\mathbf{x}) \right) \right) d\mathcal{V} \\
&= \sum_{i=1}^{N_P} \frac{\partial \hat{\varphi}_{ik}(t)}{\partial t} \int_{\Omega_k^h} \vartheta_{jk}(\mathbf{x}) \vartheta_{ik}(\mathbf{x}) d\mathcal{V} \\
&= \sum_{i=1}^{N_P} \frac{d\hat{\varphi}_{ik}(t)}{dt} \delta_{ji}.
\end{aligned} \tag{3.44}$$

where δ_{ji} is the Kronecker delta function defined as,

$$\delta_{ji} = \begin{cases} 1 & , \text{if } j = i \\ 0 & , j \neq i. \end{cases} \tag{3.45}$$

Advection Matrix (AM) It should be noted that as the variable φ the unknown variable and the velocity field \mathbf{u} is prescribed, this advection term is not a non-linear term. Taking an integration by parts yields,

$$\mathbf{AM} := \oint_{\partial\Omega_k^h} \vartheta_{jk}(\mathbf{x}) \mathbf{n}_S \cdot \mathbf{f} dS - \int_{\Omega_k^h} \mathbf{f} \cdot \nabla \vartheta_{jk}(\mathbf{x}) d\mathcal{V} \tag{3.46}$$

where \mathbf{n}_S is the normal vector to the cell border, directing outward of the cell. In the DG method, although the solution can be discontinuous across the borders of the cells, each common border of two adjacent cells must be assigned a unique flux function \mathbf{f}^* which is called the numerical flux. Such an explicit specification of the fluxes provides the conservative property of the DG method. The numerical flux is required to be substituted only into the first term of formulation (3.46) as this term lives on the cell border where the adjacent cells are in connection. In order to construct a numerical flux at a cell border, the solution needs to be reconstructed there. The solution reconstruction in the DG method is commonly made by applying either a 1st-order upwind scheme or a 2nd-order central scheme using the inner- and the outer-cell values. An upwind numerical flux can be defined as,

$$\mathbf{f}_U^* = \begin{cases} \varphi_{hp}^- \mathbf{u}_{hp}^-, & \frac{\mathbf{u}_{hp}^- \cdot \mathbf{n}_S + \mathbf{u}_{hp}^+ \cdot \mathbf{n}_S}{2} \geq 0, \\ \varphi_{hp}^+ \mathbf{u}_{hp}^+, & \frac{\mathbf{u}_{hp}^- \cdot \mathbf{n}_S + \mathbf{u}_{hp}^+ \cdot \mathbf{n}_S}{2} < 0. \end{cases} \tag{3.47}$$

The upwind scheme is appropriate for being used if the spatial differential term represents a directional phenomenon such as the advection. This is because this scheme is characterized by high dissipation and low dispersion errors. The central scheme is a proper choice for the terms representing the directionless phenomena such as the diffusion. This is because this scheme is characterized by low dissipation and high dispersion errors. Therefore, if it is applied for a directional problems, it does not dis-

sipate the non-resolved modes of the solution and advects them through the domain with wrong speeds in wrong directions, leading to the formation of the spurious oscillations, see e.g. (Marchandise et al. 2008) and (Karniadakis & Sherwin 2005). The central numerical flux can be defined as,

$$\mathbf{f}_C^* = \frac{\mathbf{u}^- \varphi_{hp}^- + \mathbf{u}^+ \varphi_{hp}^-}{2} \quad (3.48)$$

Substituting the numerical flux \mathbf{f}^* into the first term of formulation (3.46), and the DG-based representation of \mathbf{f} into the second term, yields,

$$\begin{aligned} \mathbf{AM} &:= \oint_{\partial\Omega_k^h} \vartheta_{jk}(\mathbf{x}) \mathbf{n}_S \cdot \mathbf{f}^* dS - \int_{\Omega_k^h} \left(\sum_{i=1}^{N_P} \hat{\mathbf{f}}_{ik}(t) \vartheta_{ik}(\mathbf{x}) \right) \cdot \nabla \vartheta_{jk}(\mathbf{x}) d\mathcal{V} \\ &= \oint_{\partial\Omega_k^h} \vartheta_{jk}(\mathbf{x}) \mathbf{n}_S \cdot \mathbf{f}^* dS - \left(\sum_{i=1}^{N_P} \hat{\mathbf{f}}_{ik}(t) \right) \cdot \int_{\Omega_k^h} \vartheta_{ik}(\mathbf{x}) \nabla \vartheta_{jk}(\mathbf{x}) d\mathcal{V} \end{aligned} \quad (3.49)$$

In order to explain the dispersion and dissipation characteristics of the 1^{st} -order upwind and 2^{nd} -order central schemes, we simply consider the 1D wave equation 3.32 with a wave speed $a \geq 0$. Applying the 1^{st} -order upwind scheme with $a \geq 0$ is equivalent to make a backward difference in the context of the FD method as,

$$\frac{\partial \varphi}{\partial t} + a \frac{\varphi(x_i) - \varphi(x_{i-1})}{h} = 0, \quad (3.50)$$

and applying the 2^{nd} -order central scheme is equivalent to make a central difference as,

$$\frac{\partial \varphi}{\partial t} + a \frac{\varphi(x_{i+1}) - \varphi(x_{i-1}))}{2h} = 0. \quad (3.51)$$

Considering the following Taylor's series using which the difference formulations are obtained,

$$\varphi(x) = (x - x_i) \frac{\partial \varphi(x_i)}{\partial x} + \frac{(x - x_i)^2}{2!} \frac{\partial^2 \varphi(x_i)}{\partial x^2} + \frac{(x - x_i)^3}{3!} \frac{\partial^3 \varphi(x_i)}{\partial x^3} + \dots \quad (3.52)$$

making a backward difference is like solving an equation of the form,

$$\frac{\partial \varphi}{\partial t} + a \frac{\partial \varphi}{\partial x} + b \frac{\partial^2 \varphi}{\partial x^2} = 0, \quad (3.53)$$

which has an additional dissipation term. This explained the high dissipation characteristics of the upwind scheme. On the other hand, making a central difference is like solving an equation of the form,

$$\frac{\partial \varphi}{\partial t} + a \frac{\partial \varphi}{\partial x} + c \frac{\partial^3 \varphi}{\partial x^3} = 0, \quad (3.54)$$

which although does not have a dissipation term, but has an additional term which produce extra advection. This explains the low dissipation and high dispersion characteristics of the central scheme.

Diffusion Matrix (DM) Following (Arnold et al. 2002), this 2^{nd} -order term can be converted to a couple of the 1^{st} -order terms by introducing an auxiliary variable σ_{hp} as,

$$\sigma_{hp} = \nabla \varphi_{hp} \quad (3.55a)$$

$$\mathbf{DM} := \int_{\Omega_k^h} \vartheta_{jk}(\mathbf{x}) (\nabla \cdot \sigma_{hp}) d\mathcal{V} \quad (3.55b)$$

Multiplying a set of the test vector functions $\psi_{jk}(\mathbf{x})$ to equation (3.55a) and integrating over the cell Ω_k^h yields,

$$\int_{\Omega_k^h} \psi_{jk}(\mathbf{x}) \cdot \sigma_{hp} d\mathcal{V} = \int_{\Omega_k^h} \psi_{jk}(\mathbf{x}) \cdot \nabla \varphi_{hp} d\mathcal{V} \quad (3.56)$$

Taking an integration by parts from the right hand side of equation (3.56) yields,

$$\int_{\Omega_k^h} \psi_{jk}(\mathbf{x}) \cdot \sigma_{hp} d\mathcal{V} = \oint_{\partial\Omega_k^h} \varphi^* \mathbf{n}_S \cdot \psi_{jk}(\mathbf{x}) dS - \int_{\Omega_k^h} \varphi_{hp} \nabla \cdot \psi_{jk}(\mathbf{x}) d\mathcal{V}, \quad (3.57)$$

where φ^* is a numerical flux which is defined subsequently. Taking an integration by parts from formulation (3.55b) yields,

$$\mathbf{DM} := \oint_{\partial\Omega_k^h} \vartheta_{jk}(\mathbf{x}) \mathbf{n}_S \cdot \sigma^* dS - \int_{\Omega_k^h} \sigma_{hp} \cdot \nabla \vartheta_{jk}(\mathbf{x}) d\mathcal{V} \quad (3.58)$$

where σ^* is a numerical flux which is defined subsequently. Choosing $\psi = \nabla \vartheta$, the second term in formulation (3.58) can be replaced by equation (3.57). Before doing that, an integration by parts is taken from the second term in the right hand side of equation (3.57) in order to prevent the appearance of the second derivative of ϑ . It yields,

$$\begin{aligned} \int_{\Omega_k^h} \psi_{jk}(\mathbf{x}) \cdot \sigma_{hp} d\mathcal{V} &= \oint_{\partial\Omega_k^h} \varphi^* \mathbf{n}_S \cdot \psi_{jk}(\mathbf{x}) dS \\ &\quad - \oint_{\partial\Omega_k^h} \varphi_{hp} \mathbf{n}_S \cdot \psi_{jk}(\mathbf{x}) dS + \int_{\Omega_k^h} \psi_{jk}(\mathbf{x}) \cdot (\nabla \varphi_{hp}) d\mathcal{V} \\ &= \oint_{\partial\Omega_k^h} (\varphi^* - \varphi_{hp}) \mathbf{n}_S \cdot \psi_{jk}(\mathbf{x}) dS + \int_{\Omega_k^h} \psi_{jk}(\mathbf{x}) \cdot (\nabla \varphi_{hp}) d\mathcal{V} \end{aligned} \quad (3.59)$$

Therefore,

$$\begin{aligned}
\mathbf{DM} &:= \oint_{\partial\Omega_k^h} \vartheta_{jk}(\mathbf{x}) \mathbf{n}_S \cdot \sigma^* dS - \oint_{\partial\Omega_k^h} (\varphi^* - \varphi_{hp}) \mathbf{n}_S \cdot \nabla \vartheta_{jk} dS \\
&\quad - \int_{\Omega_k^h} \nabla \vartheta_{jk} \cdot \nabla \varphi_{hp} d\mathcal{V}
\end{aligned} \tag{3.60}$$

where the numerical fluxes φ^* and σ^* can be obtained applying one of the different methods listed in (Arnold et al. 2002). For the present method, the interior penalty (IP) method is used, according to which, φ^* and σ^* can be obtained as,

$$\varphi^* = \{\varphi_{hp}\} = \frac{1}{2} (\varphi_{hp}^- + \varphi_{hp}^+) \tag{3.61}$$

and

$$\sigma^* = \{\nabla \varphi_{hp}\} - \alpha \llbracket \varphi_{hp} \rrbracket = \frac{1}{2} ((\nabla \varphi_{hp})^- + (\nabla \varphi_{hp})^+) - \alpha (\varphi_{hp}^- - \varphi_{hp}^+) \mathbf{n}_S \tag{3.62}$$

where α is a penalty parameter which is necessary for the numerical stabilization, see e.g. (Douglas & Dupont 1976). Substituting the DG-based representation of φ_{hp} into the last term of the formulation (3.60) yields,

$$\begin{aligned}
\int_{\Omega_k^h} \nabla \vartheta_{jk} \cdot \nabla \varphi_{hp} d\mathcal{V} &= \int_{\Omega_k^h} \nabla \vartheta_{jk} \cdot \nabla \left(\sum_{i=1}^{N_P} \hat{\varphi}(t) \vartheta_{ik}(\mathbf{x}) \right) d\mathcal{V} \\
&= \left(\sum_{i=1}^{N_P} \hat{\varphi}(t) \right) \int_{\Omega_k^h} \nabla \vartheta_{jk} \cdot \nabla \vartheta_{ik}(\mathbf{x}) d\mathcal{V}.
\end{aligned} \tag{3.63}$$

Source Matrix (SM) Substituting the DG-based representation of s_φ as

$$s_\varphi = \sum_{j=1}^{N_P} \hat{s}(t)_{jk} \vartheta_{jk}(\mathbf{x}) \text{ yields,}$$

$$\begin{aligned}
\mathbf{SM} &:= \int_{\Omega_k^h} \vartheta_{jk}(\mathbf{x}) \left(\sum_{i=1}^{N_P} \hat{s}_{\varphi,ik}(t) \vartheta_{ik}(\mathbf{x}) \right) d\mathcal{V} \\
&= \sum_{i=1}^{N_P} \hat{s}_{\varphi,ik}(t) \int_{\Omega_k^h} \vartheta_{ik}(\mathbf{x}) \vartheta_{jk}(\mathbf{x}) d\mathcal{V} \\
&= \sum_{i=1}^{N_P} \hat{s}_{\varphi,ik}(t) \delta_{ji}.
\end{aligned} \tag{3.64}$$

3.4.3 Gradient Calculation

The gradient of a variable φ in the DG method can be calculated in two different ways including,

- **Broken gradient:** In this method, considering the DG-based representation of the variable as $\varphi = \sum_{j=1}^{N_P} \hat{\varphi}(t)_{jk} \vartheta_{jk}(\mathbf{x})$, the gradient can be obtained by differentiating the prescribed OBPS $\vartheta_{jk}(\mathbf{x})$. Therefore, denoting the gradient by $\mathbf{G} = \nabla \varphi$, its DG-based representation can be obtained as,

$$\sum_{j=1}^{N_P} \hat{\mathbf{G}}_{jk}(t) \vartheta_{jk}(\mathbf{x}) = \sum_{j=1}^{N_P} \hat{\varphi}_{jk}(t) \nabla \vartheta_{jk}(\mathbf{x}) \quad (3.65)$$

- **By-flux gradient:** In this method, each component of the gradient vector is written in the form of a divergence as,

$$G_i = (\nabla_i) \varphi \equiv \nabla \cdot (\mathbf{e}_i \varphi), \quad i = 1, 2, 3 \quad (3.66)$$

where \mathbf{e}_i represents the standard bases. In this way, each component of the gradient term takes the form of an advection term similar to the one in equation (3.18). Therefore, the same procedure described in section 3.4.2 for making the DG-based discretization of an advection term can be followed, together with using the central scheme for the flux reconstruction. As it is shown in (Emamy 2013), this method leads to a more accurate calculation of the gradient than the former method.

3.4.4 Boundary Condition

As it was explained in section 3.4.2, the numerical flux can be reconstructed at each of the cell borders using a combination of the inner- and the outer-cell values φ_{hp}^+ and φ_{hp}^- . If the cell border is a part of the domain boundary $\partial\Omega$, the outer-cell value is not known, as there is no neighboring cell located outside the domain Ω . If a value φ_B is imposed on the boundary as a part of the problem definition (Dirichlet boundary condition), the unknown outer-cell value can be determined in two ways, including the direct method as,

$$\varphi_{hp}^+ = \varphi_B, \quad (3.67)$$

and the mirror method as,

$$\frac{\varphi_{hp}^- + \varphi_{hp}^+}{2} = \varphi_B. \quad (3.68)$$

If the value φ_B is not specified, the unknown outer-cell value may be allowed to be equal to the inner-cell value, which means a zero gradient of φ normal to the domain boundary (Homogeneous Neumann boundary condition).

3.4.5 Reference Cell

As a numerical grid consists of a set of the cells with various geometries and locations, it is impractical to construct an OBPS within each of them. Moreover, implementing a

Gaussian quadrature rule for each of the cells is a tedious task. In order to overcome this problem, one can introduce a reference element in another domain characterized by the coordinates (ξ_1, ξ_2) which are related to the coordinates (x_1, x_2) using a transformation as,

$$\mathbf{x} = \mathbf{T}_k(\boldsymbol{\xi}) = \mathbf{M}_k \boldsymbol{\xi} + \mathbf{A}_k, \quad k = 1, \dots, N_C \quad (3.69)$$

where \mathbf{M}_k is a matrix which accounts for different types of linear deformation such as stretch, rotation and shear, see (Kummer 2012). The vector \mathbf{A}_k performs the translation. As the centroid of the reference cell is commonly located on $(\xi_1 = 0, \xi_2 = 0)$, the vector \mathbf{A}_k gives the coordinates of the centroid of the cell Ω_k^h . Figure 3.8 schematically shows a linear transformation between for instance, a triangular reference cell and a triangular cell Ω_k^h . According to this transformation, a general function $f(\mathbf{x})$ can be

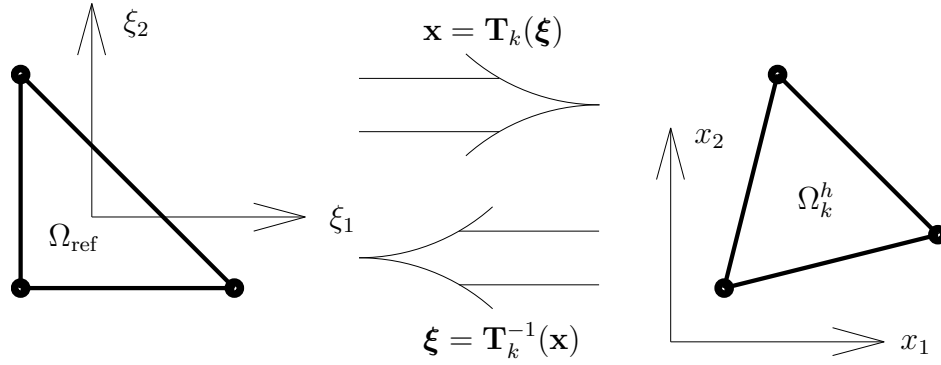


Figure 3.8 A schematic representation of a linear transformation between the reference cell Ω_{ref} and a cell Ω_k^h

transferred to its corresponding function $g(\boldsymbol{\xi})$ as,

$$f(\mathbf{T}_k(\boldsymbol{\xi})) = \frac{1}{\sqrt{\det(\mathbf{M}_k)}} g(\boldsymbol{\xi}), \quad k = 1, \dots, N_C \quad (3.70)$$

This transformation can be applied to transfer, for instance, a solution $\varphi_{hp}(\mathbf{x}, t)$ to $\varphi_{ref}(\boldsymbol{\xi}, t)$, and the OBPS $\vartheta_{jk}(\mathbf{x})$ to $\vartheta_j(\boldsymbol{\xi})$, where $k = 1, \dots, N_C$ and $j = 1, \dots, N_P$. It should be noted that the values of the coefficients $\hat{\varphi}_{jk}(t)$ stay the same in the transformation as they are only time dependent functions.

Furthermore, an integral over a cell Ω_k^h can be transferred to an integral over the reference cell Ω_{ref} as,

$$\int_{\Omega_k^h} d\mathcal{V} = \det(\mathbf{M}_k) \int_{\Omega_{ref}} d\mathcal{V}_{ref}, \quad k = 1, \dots, N_C \quad (3.71)$$

3.4.6 CFL Condition

A suitable time step used for performing an explicit time integration, should be able to fulfil two requirements. The first requirement is that the time step results a stable solution. The second requirement is that the time step corresponds to the domination of the spatial error. It means that the error should not be reduced by reducing the time step size. The CFL (Courant-Friedrichs-Lewy) condition provides an estimate for determining the time step corresponding to a stable solution. This condition can be expressed as,

$$\frac{|\mathbf{u}|_{max} \Delta t}{h} \leq \text{CFL}, \quad (3.72)$$

where $|\mathbf{u}|_{max}$ denotes the maximum speed of the propagating characteristics, Δt denotes the time step and h denotes the characteristics cell size. Accordingly, the CFL number 1 means that the characteristics of the solution pass one cell in a time step. The CFL number 1 is the maximum CFL number corresponding to a stable solution if the unknown variable has no spatial variation within each cell, such as in the FV method. But in the DG method where the inner-cell spatial variation is considered, the maximum CFL number is less than 1. Cockburn & Shu (2001) showed that using an OBPS of degree p and a Runge-Kutta method of order $p + 1$, the CFL number of $1/(2P + 1)$ leads to a stable solution. But this CFL number restricts the OBPS degree by the order of the RK method. It is shown in (Hesthaven 2008) that the asymptotic behavior of the time step which leads to stable solution is proportional to h/p^2 . Therefore, the CFL number according to their analysis is c/p^2 , where c is a constant.

3.5 The In-House Code "BoSSS"

In the present research, the corresponding numerical techniques are implemented in the context of an available in-house code named as "BoSSS" (Bounded Support Spectral Solver) which is under development in the Chair of Fluid Dynamics of Mechanical Engineering Department in Darmstadt University of Technology. This code was initiated by Kummer (2012) as the objective of his PhD research.

The main purpose of BoSSS is to solve the conservation laws, such as the one represented as the equation (3.18). This code which is programmed in C# following an object oriented paradigm, is designed as a collection of rather small software libraries. The libraries are organized in 6 layers including,

- Native: This layer includes 3rd-party codes such as MPI, Hypre, etc.
- ilPSP: The name of this layer is an abbreviation of "intermediate language Parallel Scientific Platform". This layer includes .Net wrappers for the 3rd-party codes, etc.
- L1-Platform: This layer includes utility libraries, etc.
- L2-Foundation: This layer includes libraries for spatial discretization using DG method.

- L3-Solution: This layer includes libraries for the time integration techniques, templates for the layer L4-Application, etc.
- L4-Application: This layer included several application codes such as Navier-Stokes solvers, level set method, etc. The codes in this layer use the libraries available the lower layers, providing a simple way for programming.

For instance, the main part of an application code in the layer L4-Application for solving an advection equation such as the LSA equation (2.6), can be written as,

```
{
AdvectionTerm = new SpatialOperator(1,2,1,new string[]
    {"LevelSet", "Velocity[0]", "Velocity[1]", "Result"});

SO.EquationComponents["Result"].Add(new NumericalFlux(m_app));

TimeIntegrator = new RungeKutta(Context,RungeKutta.RungeKuttaScheme.TVD3
    ,AdvectionTerm,new CoordinateMapping(LevelSet)
    ,new CoordinateMapping(Velocity[0], Velocity[1]));
}
```

where the LS function is the unknown of the equation and the prescribed velocity components are considered as parameters. More information on the software architecture of BoSSS can be found in (Kummer 2012). In addition, more technical details in developing application codes in the layer L4-Application can be found in (Emamy 2010).

3.6 Solving the Level Set Advection Equation

The spatial discretization of the advection term included in the LSA equation (2.6) can be done using the DG method as explained in the section 3.4.2. The numerical flux is reconstructed using a 1st-order upwind scheme explained in 3.4.2. Moreover, a 3rd-order TVD RK scheme, as explained in 3.3.2.2, is used for the time integration.

3.7 Solving the Level Set Re-Initialization Equation

Achieving a stable numerical solution to the LSRI equation (3.9) is not as straightforward as the LSA equation (2.6). An indigested approach for solving this equation is to consider the hyperbolic term $Sign(\phi^0)(|\nabla\phi| - 1)$ as a source term and calculating each components of $\nabla\phi$ using the DG method as explained in section 3.4.3. This method does not necessarily result in a stable numerical solution because it is not monotonicity preserving.

The behavior of the LSRI equation with respect to a numerical scheme, can be analyzed in a more illuminative way when it is considered as a Hamilton-Jacobi (HJ) equation, see e.g. (Osher & Fedkiw 2003). A general form of the HJ equation can be written for the LS function $\phi(\mathbf{x}, t)$ as,

$$\frac{\partial \phi}{\partial t} + H(\phi_x, \phi_y, \phi_z) = 0, \quad (3.73)$$

where H is called the Hamiltonian, and ϕ_{x_i} represents $\partial \phi / \partial x_i$. The difficulties in solving the HJ equation arise from the fact that this equation develops solutions with singular derivatives even if the initial condition is smooth, see e.g. (Crandall & Lions 1983). In order to overcome this problem, Crandall & Lions (1983) proposed a generalized concept of the solution which is called the viscosity solution. This name refers to the method of vanishing viscosity which is used to prove the existence of this type of solutions. The viscosity solution is a weak solution which does not need to be differentiable everywhere, while its existence, uniqueness and stability hold under certain assumptions. Therefore, the solution to every HJ equation needs to converge to a viscosity solution. Crandall & Lions (1984) proved that the class of monotone schemes (see section 3.3.2.2) results the solutions which can meet this condition. Osher & Shu (1991) investigated the use of a variety of monotone schemes for solving the HJ equations, such as the Lax-Friedrichs scheme, the Roe-Fix scheme and the Godunov's scheme, see e.g. (Osher & Fedkiw 2003). Specifically for solving the LSRI equation (3.9) as an HJ equation, Osher & Fedkiw (2003) recommended to use the Godunov's scheme, which is used in the present research accordingly.

3.7.1 Godunov's Scheme

The Godunov's scheme was applied by Bardi & Osher (1991) for solving the HJ equation. In order to describe this scheme, we consider the variables $G_i = \phi_{x_i}$ ($i = 1, 2, 3$) representing the components of $\nabla \phi$. Giving the DG-based representation of ϕ , the DG-based representation of each of the variables G_i can be obtained using the "By-Flux" method as explained in the section 3.4.3. In order to use the "By-Flux" method, one needs to calculate the numerical fluxes $f_i^* = (\mathbf{e}_i \phi)^*$ ($i = 1, 2, 3$), either by using an upwind method or a downwind method as,

$$f_i^{*U} = \begin{cases} \phi^- \mathbf{e}_i, & \mathbf{e}_i \cdot \mathbf{n}_s \geq 0 \\ \phi^+ \mathbf{e}_i, & \mathbf{e}_i \cdot \mathbf{n}_s < 0 \end{cases} \quad (3.74)$$

$$f_i^{*D} = \begin{cases} \phi^- \mathbf{e}_i, & \mathbf{e}_i \cdot \mathbf{n}_s < 0 \\ \phi^+ \mathbf{e}_i, & \mathbf{e}_i \cdot \mathbf{n}_s \geq 0 \end{cases} \quad (3.75)$$

where ($i = 1, 2, 3$), and ϕ^- and ϕ^+ denote the inner- and outer-cell values of ϕ at the border of a cell. In addition, \mathbf{e}_i represents the standard bases and \mathbf{n}_s denotes the normal vector to the border of a cell. Following (Yan & Osher 2011), the variables G_i^U and G_i^D ($i = 1, 2, 3$), are introduced using the numerical flux f_i^{*U} or f_i^{*D} respectively, for calculating G_i . In the Godunov's scheme, the Hamiltonian is approximated by a numerical Hamiltonian \tilde{H} as a function of G_i^U and G_i^D , which can be expressed as,

$$\begin{aligned}
H(G_1, G_2, G_3) &\approx \tilde{H}(G_1^U, G_1^D, G_2^U, G_2^D, G_3^U, G_3^D) \\
&= \text{ext}_x \text{ext}_y \text{ext}_z H(G_1, G_2, G_3),
\end{aligned} \tag{3.76}$$

where

$$\begin{aligned}
\text{ext}_x H(G_1, G_2, G_3) &= \begin{cases} \min(H(G_1^U, G_2, G_3), H(G_1^D, G_2, G_3)), & G_1^U \leq G_1^D \\ \max(H(G_1^U, G_2, G_3), H(G_1^D, G_2, G_3)), & G_1^U > G_1^D \end{cases} \\
\text{ext}_y H(G_1, G_2, G_3) &= \begin{cases} \min(H(G_1, G_2^U, G_3), H(G_1, G_2^D, G_3)), & G_2^U \leq G_2^D \\ \max(H(G_1, G_2^U, G_3), H(G_1, G_2^D, G_3)), & G_2^U > G_2^D \end{cases} \\
\text{ext}_z H(G_1, G_2, G_3) &= \begin{cases} \min(H(G_1, G_2, G_3^U), H(G_1, G_2, G_3^D)), & G_3^U \leq G_3^D \\ \max(H(G_1, G_2, G_3^U), H(G_1, G_2, G_3^D)), & G_3^U > G_3^D \end{cases}
\end{aligned}$$

The numerical Hamiltonian 3.76 is then considered as a source term in the equation and the equation is solved by performing a time integration.

3.7.2 Godunov's Scheme for Solving the Level Set Re-initialization Equation

The term $Sign(\phi^0)(|\nabla\phi| - 1)$ in the LSRI equation (3.9) corresponds to the Hamiltonian in the HJ equation (3.73). In order to apply the Godunov's scheme (3.76) for solving the LSRI equation, Peng et al. (1999) used a compact form which can be expressed in 2D as,

$$\begin{aligned}
&Sign(\phi^0)(|\nabla\phi| - 1) \\
&\approx \begin{cases} Sign(\phi^0)(\sqrt{\max[(G_1^{U+})^2, (G_1^{D-})^2] + \max[(G_2^{U+})^2, (G_2^{D-})^2]} - 1), & Sign(\phi^0) \geq 0 \\ Sign(\phi^0)(\sqrt{\max[(G_1^{U-})^2, (G_1^{D+})^2] + \max[(G_2^{U-})^2, (G_2^{D+})^2]} - 1), & Sign(\phi^0) < 0 \end{cases}
\end{aligned} \tag{3.77}$$

where

$$\begin{aligned}
G_i^{U+} &= \max(G_i^U, 0), \quad G_i^{U-} = \min(G_i^U, 0), \\
G_i^{D+} &= \max(G_i^D, 0), \quad G_i^{D-} = \min(G_i^D, 0),
\end{aligned} \tag{3.78}$$

and ($i = 1, 2$).

A 3rd-order TVD RK scheme, as explained in the section 3.3.2.2, is used for the time integration of this equation.

3.8 Solving the Multiphase Formulation of the Navier-Stokes Equation

The solution to the single-phase formulation of the incompressible NS equation was successfully implemented in BoSSS by Emamy (2013) as a part of her PhD research. This solver is used in the present research as a basis for implementing the solution to the two-phase formulation of the incompressible NS equation.

Representing the jumps of the density and viscosity functions $\rho(\phi)$ and $\mu(\phi)$ across the interface in terms of an OBPS of higher degree may produce spurious spatial oscillations referred as the Gibbs phenomenon. In order to resolve this problem, one needs to use a smoothed approximation of the Heaviside function employed for the mathematical representation of the density and viscosity functions, i.e. the equations (2.19) and (2.20), respectively. In the present research following (Osher & Fedkiw 2003), the formulation (4.5.1) is used to approximate the Heaviside function (2.21). This approach is referred as diffuse interface approach in the literature, see e.g. (Anderson et al. 1998), although no physical diffusion between the phases is considered in the present research. On the other hand, although this approximation may resolve the problem of the spatial oscillations, but the smoothed variations of the density and viscosity fields are not considered in the derivation of the two-phase momentum equation (2.28). In order to overcome this issue, in the present research we replace the density and viscosity functions $\rho(\phi)$ and $\mu(\phi)$ by their cell-averaged values $\bar{\rho}(\phi)$ and $\bar{\mu}(\phi)$, respectively. As in the DG method the equations are solved locally in cell-wise manner, using the cell-averaged density and viscosity provides constant distributions of these fluid properties within the local domain of solution. This in turn provides the possibility of taking the density and viscosity functions out of the spatial differentiations. Therefore, the momentum equation (2.28) is replaced by the following equation,

$$\frac{\partial \mathbf{u}}{\partial t} + \nabla \cdot (\mathbf{u}\mathbf{u}) = -\frac{\nabla P}{\bar{\rho}(\phi)} + \frac{1}{Re} \frac{\bar{\mu}(\phi)}{\bar{\rho}(\phi)} \nabla^2 \mathbf{u} + \frac{\sigma \kappa \mathbf{n} \delta_{\mathcal{I}}}{\bar{\rho}(\phi) We} + \frac{\mathbf{e}_g}{Fr^2}. \quad (3.79)$$

In the context of the code BoSSS, using the cell-averaged values of a variable is equivalent to use an OBPS of degree zero for the DG-based representation of that variable. Following (Emamy 2013), we solve the system of two-phase incompressible NS equations (2.27) and (3.79) employing an algorithm proposed by Karniadakis et al. (1991). This algorithm uses a mixed explicit-implicit stiffly stable time integration scheme together with a splitting scheme for decoupling the velocity and pressure fields. The order of time integration used in the present research is 1. Following this algorithm, the momentum equation (3.79) is split into three equations and the time integration over $[t_n, t_{n+1}]$ is performed in three stages as follows:

- Advection part:

$$\frac{\gamma_0 \hat{\mathbf{u}} - \alpha_1 \mathbf{u}^n}{\Delta t} = \beta_1 \left[-\nabla \cdot (\mathbf{u}^n \mathbf{u}^n) + \frac{\sigma \kappa^n \mathbf{n}^n \delta(\phi^n)}{\rho(\phi^n) We} + \frac{\mathbf{e}_y}{Fr^2} \right], \quad (3.80)$$

where the superscript n of a variable implies that variable at the time $t = t_n$. The variables with the superscript n are considered as the known variables. Moreover, the constant values $\gamma_0 = \alpha_0 = \beta_0 = 1$ are the coefficients corresponding to the 1st-order time integration. The spatial terms of this equation, including the advection term and the source term, are discretized as described in the section 3.4.2. This equation is solved explicitly for obtaining the velocity field $\hat{\mathbf{u}}$ which is not necessarily divergence free.

- Pressure part:

$$\gamma_0 \frac{\hat{\mathbf{u}} - \mathbf{u}}{\Delta t} = - \frac{\nabla P^{n+1}}{\bar{\rho}(\phi^n)} \quad (3.81)$$

where the velocity field $\hat{\mathbf{u}}$ is supposed to be divergence free. As this step of the algorithm consists of projecting the non-divergence free velocity field \mathbf{u} to a divergence free velocity field $\hat{\mathbf{u}}$, it is called the projection step. The pressure field which results in the divergence free velocity field $\hat{\mathbf{u}}$, is obtained by solving a Poisson's equation before solving the equation (3.81). This Poisson's equation which is obtained by taking a divergence from the equation (3.81) together with the condition $\nabla \cdot \hat{\mathbf{u}} = 0$, is as follows:

$$-\nabla^2 P^{n+1} = -\gamma_0 \bar{\rho}(\phi^n) \frac{\nabla \cdot \mathbf{u}}{\Delta t}, \quad (3.82)$$

where the 2nd-order spatial differentiation is discretized as described in the section 3.4.2.

- Viscous term:

$$\gamma_0 \frac{\mathbf{u}^{n+1} - \hat{\mathbf{u}}}{\Delta t} = \frac{\bar{\mu}(\phi^n)}{Re \bar{\rho}(\phi^n)} \nabla^2 \mathbf{u}^{n+1}, \quad (3.83)$$

which is solved implicitly for finally obtaining the velocity field \mathbf{u}^{n+1} . The 2nd-order spatial differentiation is discretized as described in the section 3.4.2.

The reason why the corresponding source terms of the surface tension and the gravity are included in the advection part, is that these terms are expected to add additional divergence to the velocity field. Therefore, they are included before applying the projection step.

As in order to reduce the computational cost the LSRI equation (3.9) is solved only in a narrow band around the interface, the LS function is only affected in this region resulting in a jump across the border of the narrow band. This jump may produce spurious spatial oscillations if the LSA equation (2.6) is solved in terms of this SDLS function. In order to overcome this issue, we consider a secondary LS function which is not re-initialized but represents the same interface. The SDLS function is denoted by ϕ and the NSDLS function is denoted by ϕ_{NSD} . Therefore, the interface movement is simulated by solving the LSA equation (2.6) in terms of this NSDLS function. Then at each time step, the DG field of the NSDLS function is copied to the DG field of the SDLS function and re-initialized in the narrow band. This SDLS function is employed for calculating the interface curvature and constructing the density and vis-

cosity fields at the same time step.

Finally, the entire procedure of simulating an immiscible two-phase flow can be summarized as follows:

1. Copying the DG field of the NSDLS function ϕ_{NSD}^n to the DG field of the SDLS function ϕ^n .
2. Solving the LSRI equation (3.9) for the SDLS function ϕ^n in a narrow band around the interface.
3. Calculating the interface normal vector \mathbf{n}^n using the expression (2.13) and consequently the interface curvature κ^n using the expression (2.14), in terms of the the SDLS function ϕ^n .
4. Constructing the density and viscosity fields using in terms of the SDLS function ϕ^n using the expressions (2.19) and (2.20), respectively.
5. Solving the two-phase NS equations (2.27) and (3.79) as described in the present section, for obtaining the velocity field \mathbf{u}^{n+1} and the pressure field P^{n+1} .
6. Solving the LSA equation (2.6) for obtaining the the NSDLS function ϕ_{NSD}^{n+1} having the field of the NSDLS function ϕ_{NSD}^n and velocity field \mathbf{u}^{n+1} .

4 Numerical Simulations and Results

4.1 Error Calculation

The numerical results of the test cases considered in the present research are analyzed based on measuring three kinds of the errors including "volume/area loss", "interface \mathbb{L}^1 -error" and " \mathbb{L}^2 -error", which are briefly describes as follows:

4.1.1 Volume/Area Loss

This error gives the difference between the volume/area occupied by the computed interface and the correct volume/area. Supposing that the region occupied by the interface corresponds to the negative part of the LS function, the volume/area of this region can be calculated as,

$$\Omega_{(\phi_{hp} < 0)}^h = \int_{\Omega_{(\phi_{hp} < 0)}^h} d\mathcal{V} \quad (4.1)$$

Therefore the volume/area loss can be calculated in percentage as,

$$\text{Area Loss} = 100 \times \frac{\Omega_{(\phi^{ref} < 0)} - \Omega_{(\phi_{hp} < 0)}^h}{\Omega_{(\phi^{ref} < 0)}} \quad (4.2)$$

where ϕ^{ref} denotes a reference LS function which is compared to the calculated field of the LS function.

4.1.2 Interface \mathbb{L}^1 -Error

This error gives an \mathbb{L}^1 measure on the spurious movement of the interface. This error can be calculated as,

$$\mathbb{L}_{\mathcal{I}}^1 = \frac{1}{\mathcal{L}} \int_{\Omega^h} |\mathcal{H}(\phi_{hp}) - \mathcal{H}(\phi^{ref})| d\mathcal{V} \quad (4.3)$$

where \mathcal{L} is the interface circumference, ϕ^{ref} is the reference level set function and $\mathcal{H}(\phi)$ is the Heaviside function. The interface \mathbb{L}^1 -Error as well as the area loss are calculated in the present research by taking a brute-force integration performing a five-stage cell division.

4.1.3 \mathbb{L}^2 Error

This error gives an \mathbb{L}^2 measure on the accuracy of computing a the field of a variable such as the LS function ϕ . This error can be calculated in two ways which make slightly different values. If the reference field of variable is available as an analytic function, the \mathbb{L}^2 error is calculated as,

$$\mathbb{L}_{\phi_{hp}}^2 = \|\phi_{hp} - \phi^{ref}\|_{\Omega^h} = \left(\int_{\Omega^h} \left[\left(\sum_{k=1}^{N_C} \sum_{j=1}^{N_P} \hat{\phi}_{jk} \vartheta_{jk} \right) - \phi^{ref} \right]^2 d\mathcal{V} \right)^{\frac{1}{2}} \quad (4.4)$$

where ϕ^{ref} is evaluated at the quadrature points in the procedure of the numerical integration. The errors in the present chapter are calculated in this way. If the reference field of the variable is available as a DG field, the first step is to subtract the computed DG field from the reference one as,

$$\begin{aligned} \phi_{hp} - \phi_{hp}^{ref} &= \sum_{j=1}^{N_P} \hat{\phi}_{jk} \vartheta_{jk} - \sum_{j=1}^{N_P} \hat{\phi}_{jk}^{ref} \vartheta_{jk} = \sum_{j=1}^{N_P} (\hat{\phi}_{jk} - \hat{\phi}_{jk}^{ref}) \vartheta_{jk} \\ &= \sum_{j=1}^{N_P} \hat{\psi}_{jk} \vartheta_{jk} = \psi \\ k &= 1, \dots, N_C \end{aligned} \quad (4.5)$$

where ϕ_{hp}^{ref} denotes the reference DG field of the variable. The next step is to calculate the \mathbb{L}^2 norm of ψ as,

$$\|\psi\| = \left\| \sum_{k=1}^{N_C} \sum_{j=1}^{N_P} \hat{\psi}_{jk} \vartheta_{jk} \right\| = \left(\int_{\Omega^h} \left(\sum_{k=1}^{N_C} \sum_{j=1}^{N_P} \hat{\psi}_{jk} \vartheta_{jk} \right)^2 d\mathcal{V} \right)^{\frac{1}{2}} \quad (4.6)$$

As ϑ_{jk} represents an OBPS, according to the Parseval's theorem one can write the final form of the \mathbb{L}^2 error as,

$$\begin{aligned} \mathbb{L}_{\phi_{hp}}^2 &= \left(\int_{\Omega^h} \sum_{k=1}^{N_C} \sum_{j=1}^{N_P} \hat{\psi}_{jk}^2 d\mathcal{V} \right)^{\frac{1}{2}} = \left(\int_{\Omega^h} \sum_{k=1}^{N_C} \sum_{j=1}^{N_P} (\hat{\phi}_{jk} - \hat{\phi}_{jk}^{ref})^2 d\mathcal{V} \right)^{\frac{1}{2}} \\ &= \left(\int_{\Omega^h} \sum_{k=1}^{N_C} \sum_{j=1}^{N_P} (\langle \phi, \vartheta_{jk} \rangle - \langle \phi^{ref}, \vartheta_{jk} \rangle)^2 d\mathcal{V} \right)^{\frac{1}{2}} \end{aligned} \quad (4.7)$$

4.2 Visualization

Apart from the error calculations performed in BoSSS, the postprocessing tasks such as generating the contour plots are done in the software VisIt. This software is a free interactive parallel visualization and graphical analysis tool for viewing scientific

data on Unix and PC platforms, see (U.S. Department of Energy 2013). In order to generate a contour plot, VisIt makes a linear interpolation between the discrete values. The inner-cell variations are not included in the BoSSS output files and these files are constructed by evaluating the variables at the cell vertices. Therefore, if the grid is coarse, making a linear interpolation between the values on the cell vertices can not generate a representative contour plot, such as the figure 4.1a which is aimed to represent an arc as the zero iso-value of an LS function. The OBPS degree $p = 3$ is used with the grid resolution $N_C = 20 \times 20$. In order to overcome this problem, it is possible in BoSSS to perform a multistage division of the cells in the procedure of constructing the output files. For instance, the effects of performing One-stage division, two-stage division and three-stage division of the cells are demonstrated in figures 4.1b, 4.1c and 4.1d, respectively. The curve shown in figure 4.1d looks quite smooth to the eye.

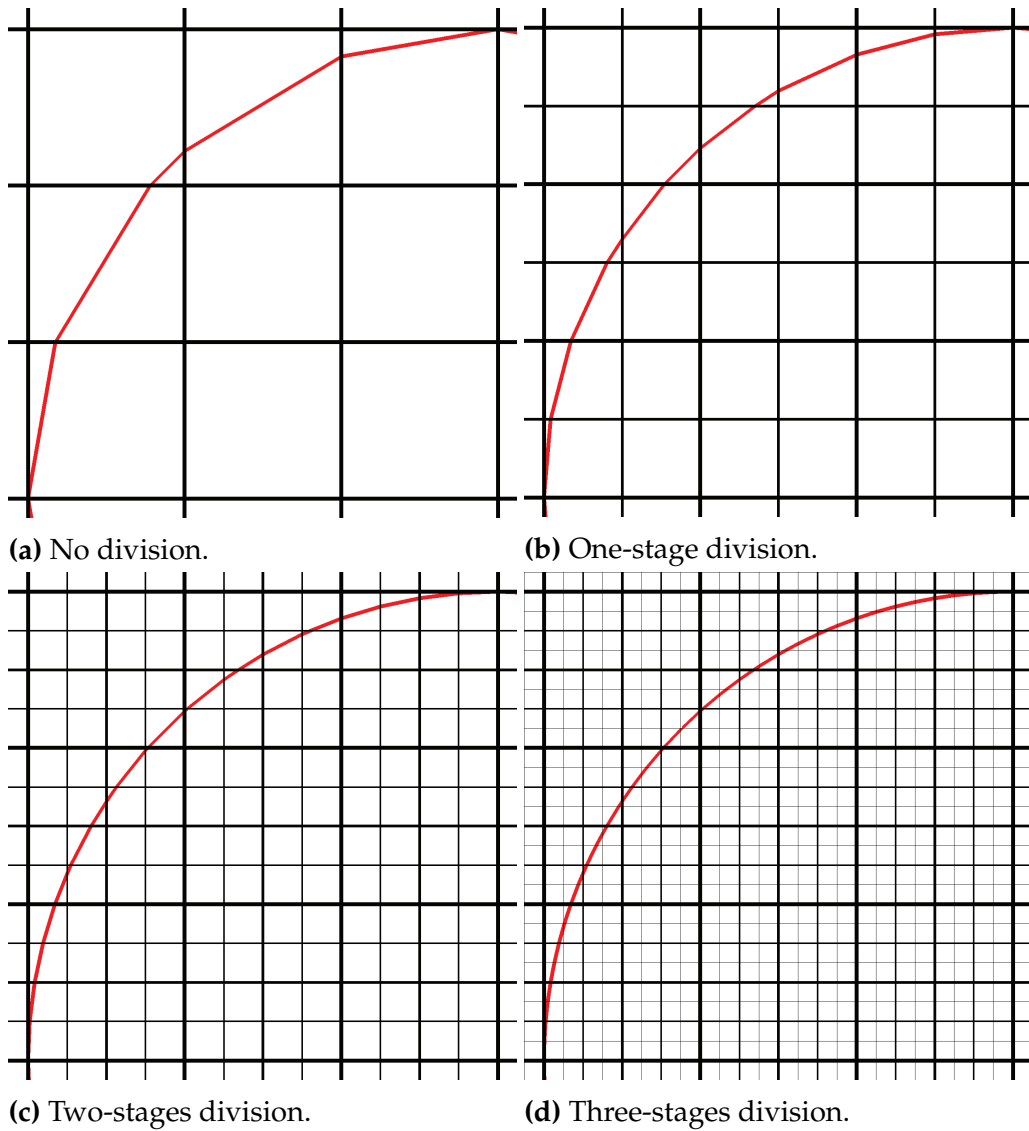


Figure 4.1 Multistage division of the cells for making the inner-cell evaluations. It is aimed to show an arc as the zero iso-surface of an LS function. The OBPS degree $p = 3$ is used with the grid resolution $N_C = 20 \times 20$.

4.3 Solutions to the Level Set Advection Equation

This section includes a number of test cases considered in order to verify the methods used in the present research for solving the LSA equation (2.6).

4.3.1 Rigid Body Rotation of a Circle

This section is assigned to verify the numerical solution to the LSA equation (2.6) by simulating the motion of an eccentric circle in a prescribed velocity field corresponding to a rigid body rotation. The role of the dissipation error in reducing the accuracy of the LS method is illustrated in this test case instructively.

Problem Description The domain of computation is a square with the lower-left corner located at $(0, 0)$ and the upper-right corner located at $(100, 100)$. The geometry of the interface is a circle with the radius $R = 15$, initially centered at $(x_c = 50, y_c = 75)$. The corresponding SDLS function of the interface can be analytically expressed as,

$$\phi^0(\mathbf{x}) = \sqrt{(x - x_c)^2 + (y - y_c)^2} - R, \quad (4.8)$$

which has singular first derivative at $(x = x_c, y = y_c)$. The three-dimensional (3D) plot of this function over the (x, y) -plane looks like a cone. The prescribed velocity field is defined as,

$$\mathbf{u}(\mathbf{x}) = u_x(x, y)\mathbf{e}_x + u_y(x, y)\mathbf{e}_y = (\pi/314)(50 - y)\mathbf{e}_x + (\pi/314)(x - 50)\mathbf{e}_y, \quad (4.9)$$

according to which, every revolution is completed after 628 unit time steps. A vector plot of this field is shown in figure 4.2. The simulation is performed until the time 62800 corresponding to 100 revolutions of the circle.

Numerical Settings The domain is discretized to a set of the quadrilateral cells with $N_C = 20 \times 20$, according to which, the circle passes almost six cells along its diameter. The reason of adopting this resolution is keeping the interface away from the cells including the apex of the cone which can not be projected over the OBPS properly. The OBPS degree is set to $p = 1, 2, 3$. The time step is set to $\Delta t = 1$. In order to verify the advantage of using an OBPS with a higher degree over the one with a lower degree but with the same N_{DoF} , a case is considered with $p = 1$ and $N_C = 37 \times 37$. The value of N_{DoF} for this case is almost equal to the case with $p = 3$ and $N_C = 20 \times 20$. The value of N_{DoF} in each cell is the number of the orthonormal basis polynomials corresponding to a certain degree of the OBPS. This value is calculated using the expression 3.37. A homogeneous Neumann boundary condition is imposed on the entire boundary.

Results Table 4.1 lists the area losses resulted by projecting the initial LS function to the OBPSs of degrees $p = 1, 2, 3$. As it is stated in the table, using $p = 1$ together

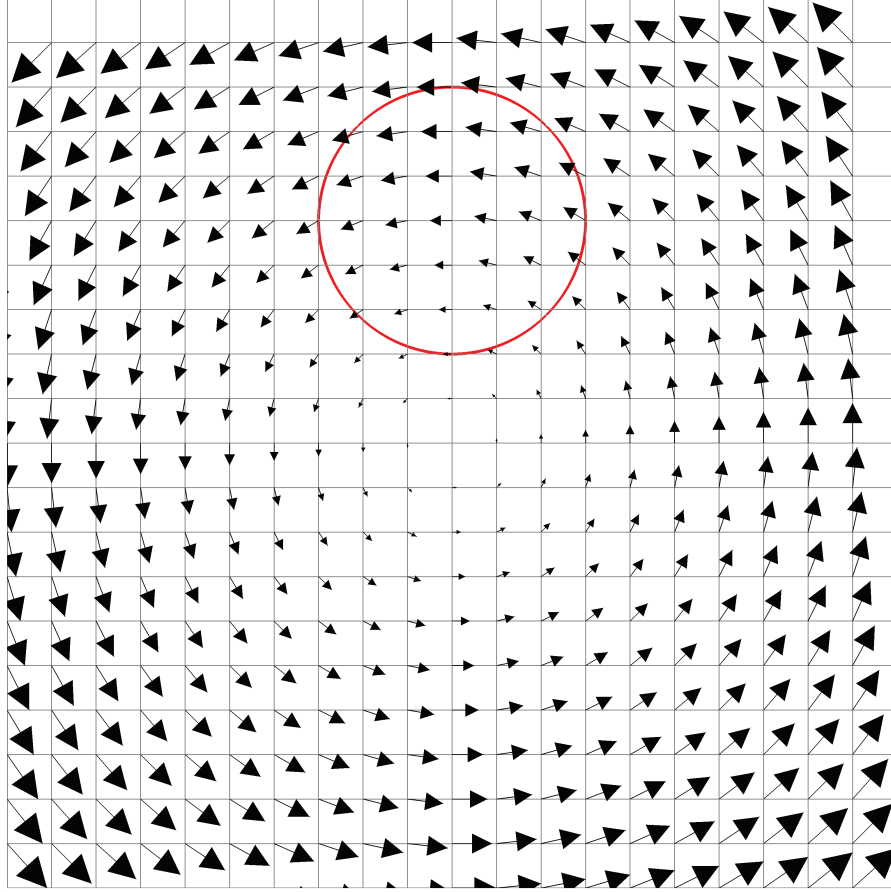


Figure 4.2 Rigid Body Rotation of a Circle: The prescribed velocity vector field.

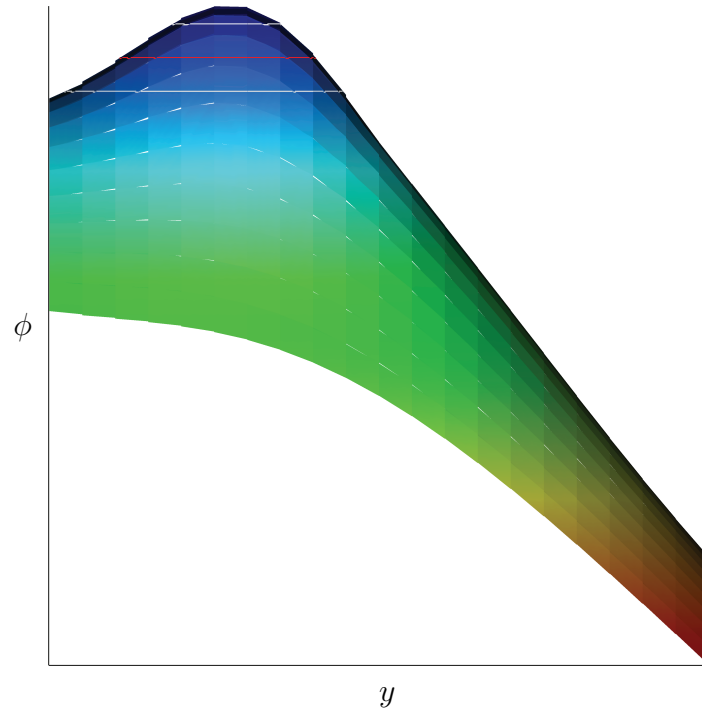
with $N_C = 20 \times 20$ results an area loss which has almost the same value of the area gain resulted by using $p = 2$ together with $N_C = 20 \times 20$. But using $p = 3$ together with $N_C = 20 \times 20$ results an area loss of one order of magnitude less. Furthermore, it is stated that although the value of N_{DoF} corresponding to the use of $p = 1$ together with $N_C = 37 \times 37$ is equal to the one corresponding to the use of $p = 3$ together with $N_C = 20 \times 20$, such level of the grid refinement can not compensate the inaccuracy imposed due to the use of $p = 1$. Figures 4.3, 4.4 show the 3D plots of the LS function in a view along the x -axis after 10 and 100 revolutions of the interface using different OBPS degrees and grid resolutions. The pictures are colored based on the values returned by the LS function. As these values are not of interest outside the interface, the color legends are excluded. These figures are aimed to illustrate the effects of the OBPS degree on the dissipation error. As it is shown in the figures, a major effect of the dissipation error is rounding the sharp apex of the cone representing the LS function. Furthermore, the dissipation error reduces the amplitude of the variation of the LS function over the domain. It is shown that the dissipation error produced by using $p = 1$ is much higher than the one produced by $p = 2$ or $p = 3$, even by making a grid refinement from $N_C = 20 \times 20$ to $N_C = 37 \times 37$. Figure 4.5 shows the shapes of the interface after 10, 20, 30, 40, 50, 60, 70, 80, 90 and 100 revolutions, using different OBPS degrees and grid resolutions. As it is shown in figure 4.5a, as a result of an extreme dissipation due to the use of $p = 1$ together with $N_C = 20 \times 20$, the interface

Table 4.1

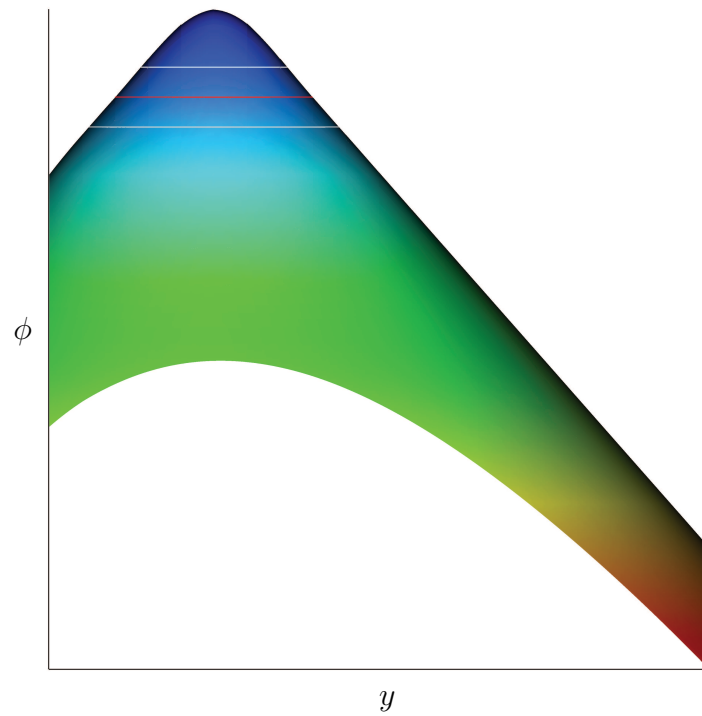
Rigid Body Rotation of a Circle: Area losses resulted by projecting the initial LS function to the OBPSs of 1, 2 and 3.

p	N_C	N_{DoF}	Area	Area Loss (%)
Exact			706.858	...
1	20×20	1200	707.105	-0.0349
2	20×20	2400	706.628	0.0325
3	20×20	4000	706.841	0.00241
1	37×37	4107	706.934	-0.0108

disappears after 40 revolutions and takes another pattern. Figure 4.5d shows that the area error is reduced by performing the grid refinement. As it is shown in figures 4.5b and 4.5c, the area error is dramatically reduced by increasing the OPBS degree. Figure 4.5c shows that there is almost no area error after 100 revolutions of the interface by using $p = 3$ together with $N_C = 20 \times 20$. Diagrams 4.6 and 4.7 demonstrate the area error as the absolute of the area loss, and an \mathbb{L}^1 measure of the interface error in 100 revolutions of the interface using different OBPS degrees and grid resolutions.

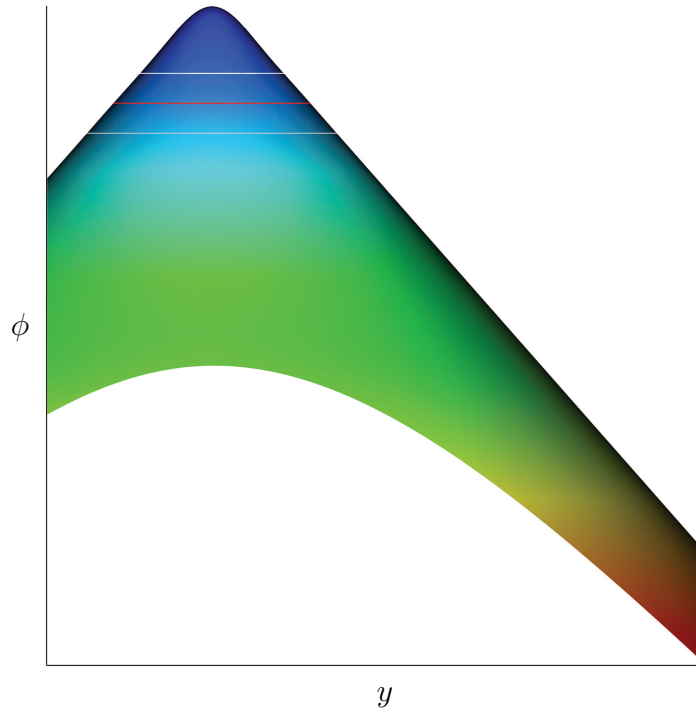


(a) $p = 1, N_C = 20 \times 20$

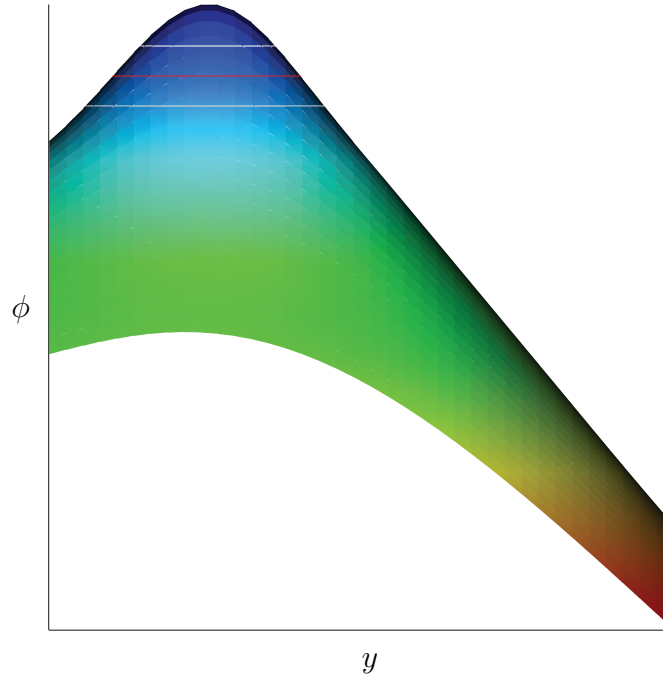


(b) $p = 2, N_C = 20 \times 20$

Figure 4.3 Rigid Body Rotation of a Circle: A side view of the 3D plot of the LS function after 10 revolutions of the interface. The red curve represents $\phi = 0$ and the white curves represent $\phi = -4$ and $\phi = 4$, respectively.

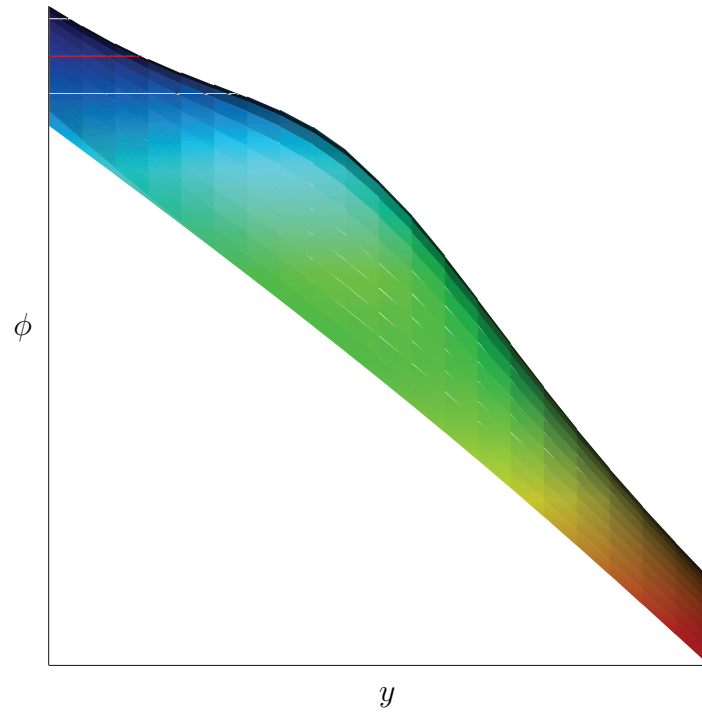


(c) $p = 3, N_C = 20 \times 20$

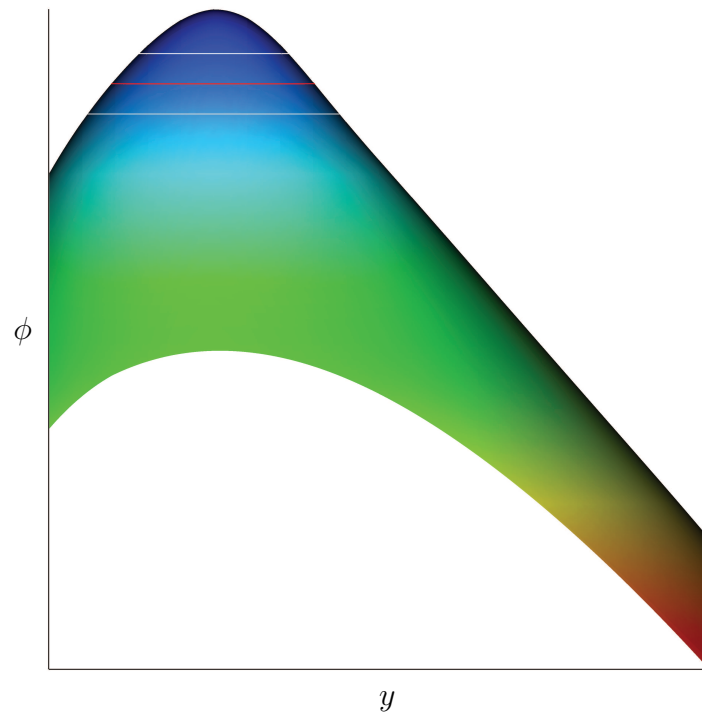


(d) $p = 1, N_C = 37 \times 37$

Figure 4.3 Rigid Body Rotation of a Circle: A side view of the 3D plot of the LS function after 10 revolutions of the interface. The red curve represents $\phi = 0$ and the white curves represent $\phi = -4$ and $\phi = 4$, respectively.

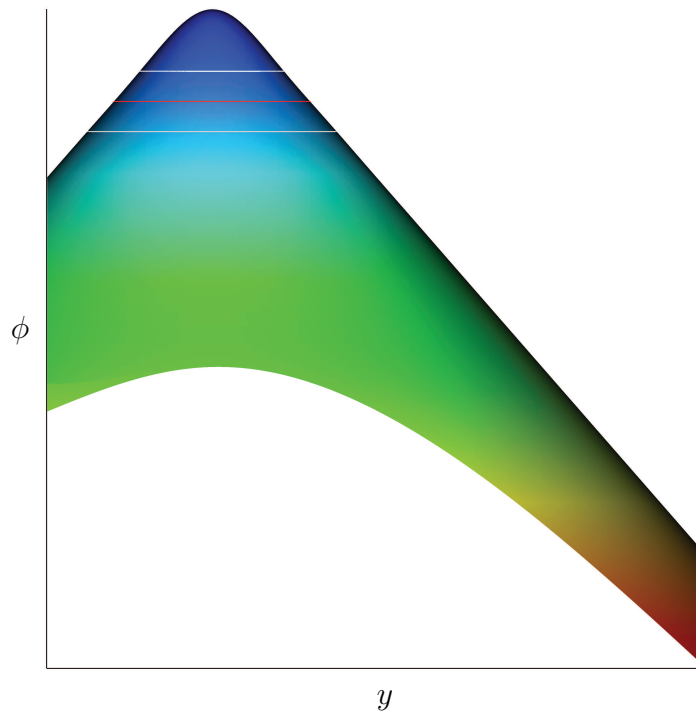


(a) $p = 1, N_C = 20 \times 20$

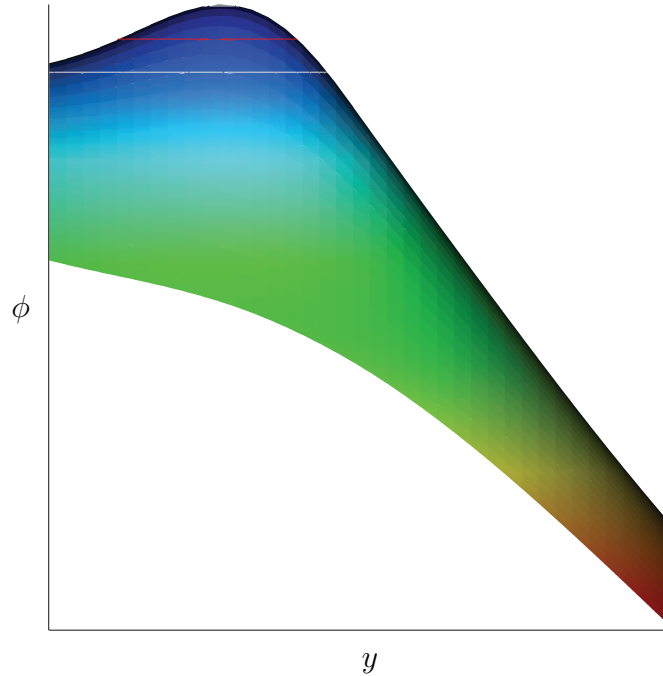


(b) $p = 2, N_C = 20 \times 20$

Figure 4.4 Rigid Body Rotation of a Circle: A side view of the 3D plot of the LS function after 100 revolutions of the interface. The red curve represents $\phi = 0$ and the white curves represent $\phi = -4$ and $\phi = 4$, respectively.

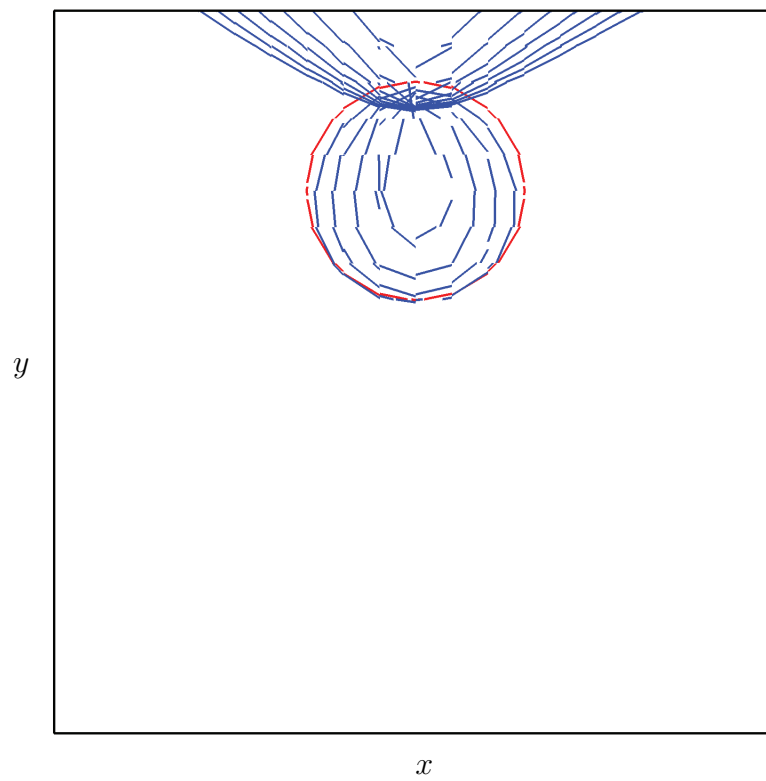


(c) $p = 3, N_C = 20 \times 20$

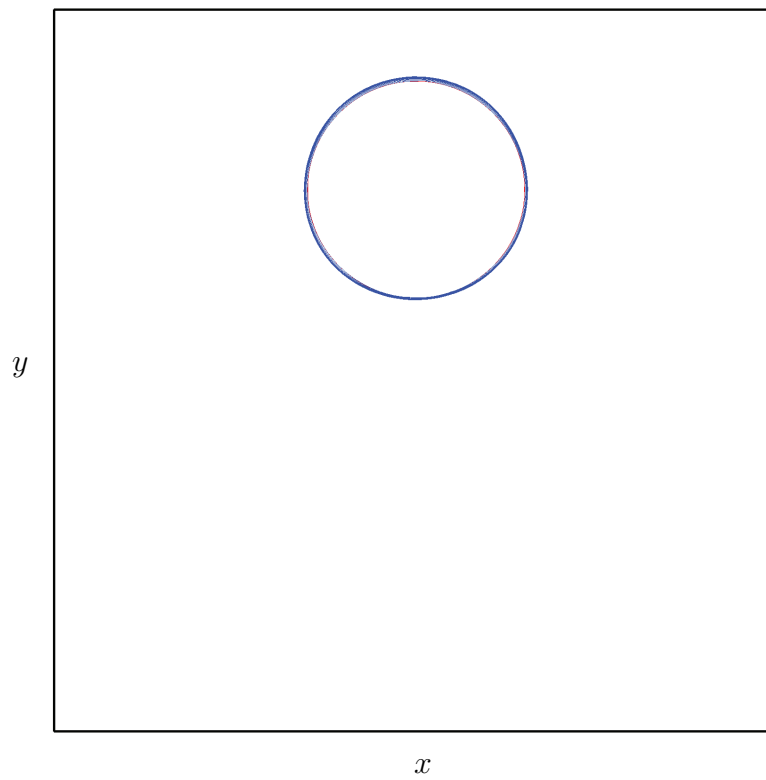


(d) $p = 1, N_C = 37 \times 37$

Figure 4.4 Rigid Body Rotation of a Circle: A side view of the 3D plot of the LS function after 100 revolutions of the interface. The red curve represents $\phi = 0$ and the white curves represent $\phi = -4$ and $\phi = 4$, respectively.

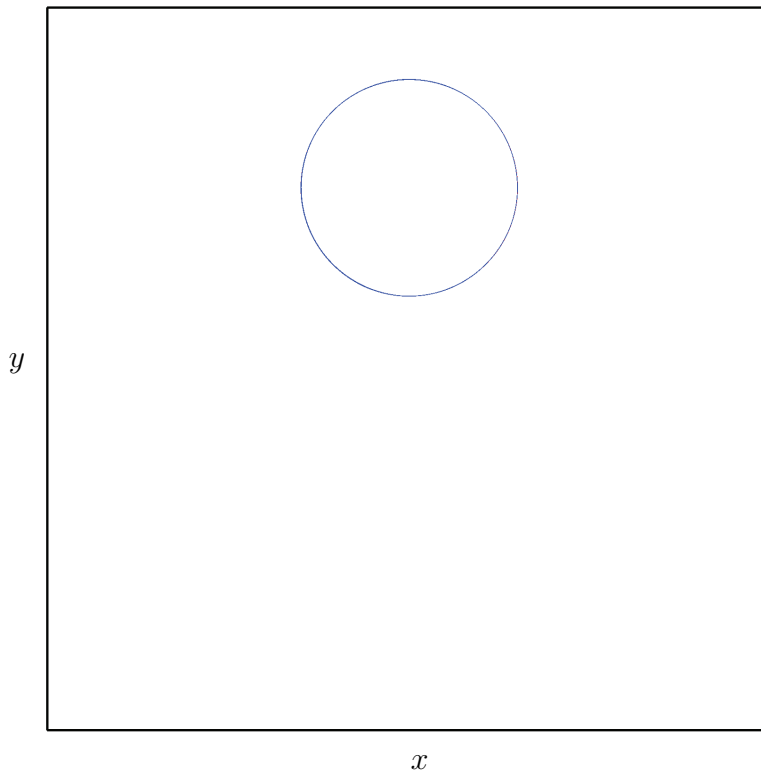


(a) $p = 1, N_C = 20 \times 20$

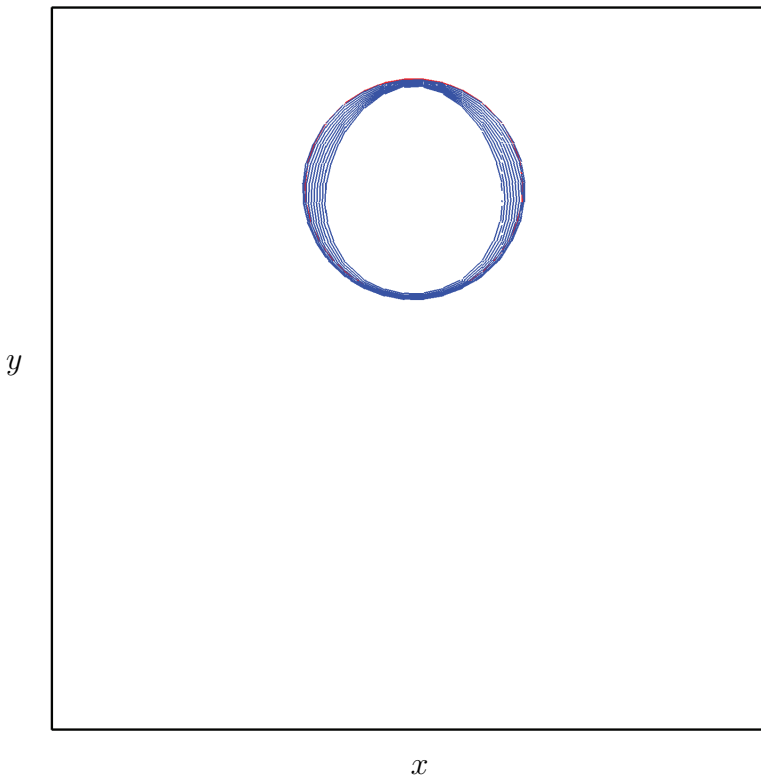


(b) $p = 2, N_C = 20 \times 20$

Figure 4.5 Rigid Body Rotation of a Circle: Shape of the interface after 10, 20, 30, 40, 50, 60, 70, 80, 90 and 100 revolutions. The red curve represents ϕ^0 .



(c) $p = 3, N_C = 20 \times 20$



(d) $p = 1, N_C = 37 \times 37$

Figure 4.5 Rigid Body Rotation of a Circle: Shape of the interface after 10, 20, 30, 40, 50, 60, 70, 80, 90 and 100 revolutions. The red curve represents ϕ^0 .

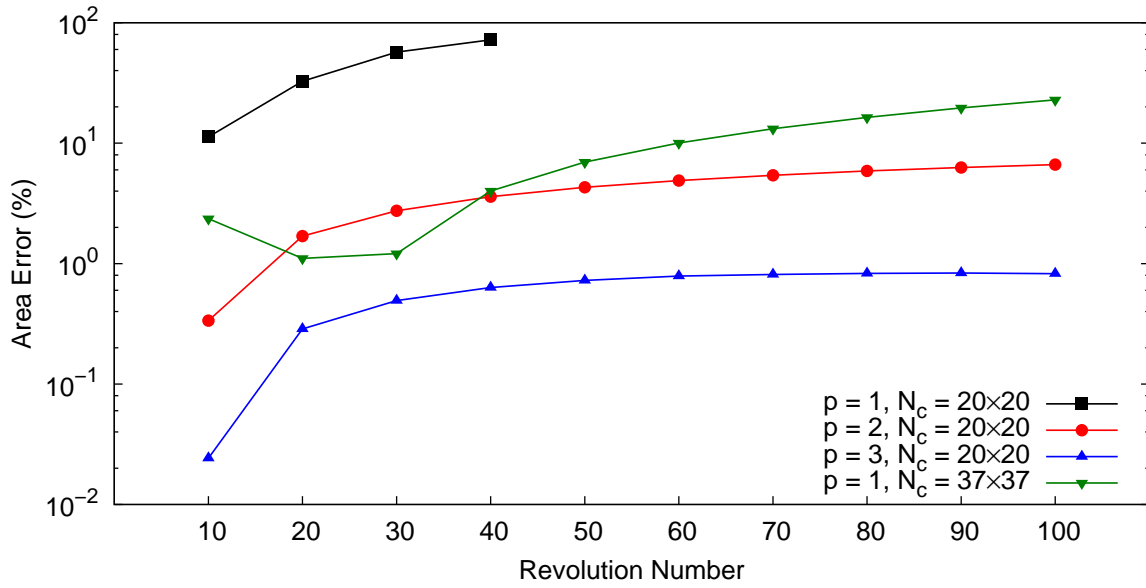


Figure 4.6 Rigid Body Rotation of a Circle: Area error produced in 100 revolutions of the interface

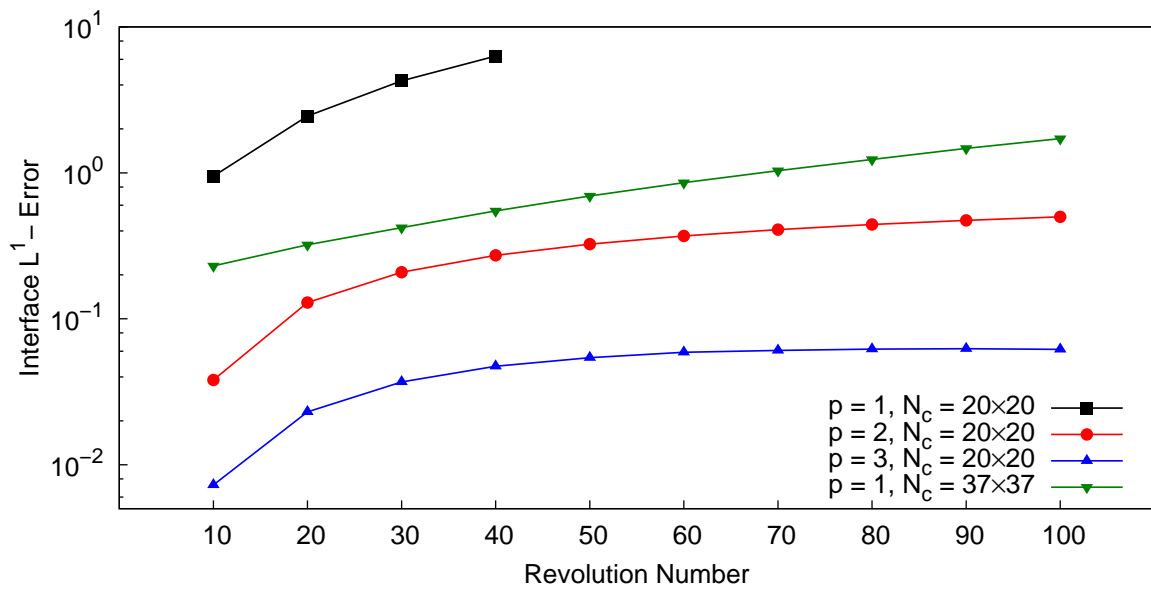


Figure 4.7 Rigid Body Rotation of a Circle: Interface L^1 -error produced in 100 revolutions of the interface

4.3.2 Rigid Body Rotation of a Slotted Disk

This section is assigned to verify the numerical solution to the LSA equation (2.6) by simulating the motion of an eccentric slotted disk in a prescribed velocity field corresponding to a rigid body rotation. This test case was originally considered by Zalesak (1979) and has been adopted as a standard benchmark in the corresponding research field. Comparing to the test case 4.3.1, the interface in this test case has a rather complex geometry intensifying the role of the dissipation error in reducing the accuracy of the LS method. Moreover, the difficulties in constructing the initial SDLS function of such a geometry is indicated in this test case.

Problem Description The domain of computation is a square with the lower-left corner located at $(0, 0)$ and the upper-right corner located at $(100, 100)$. The geometry of the interface is the border of a slotted disk with the radius $R = 15$, initially centered at $(x_c = 50, y_c = 75)$. The slot length is $L_{Slot} = 25$ and the slot width is $W_{Slot} = 5$. The interface is shown in figure 4.8. In order to construct the corresponding SDLS function

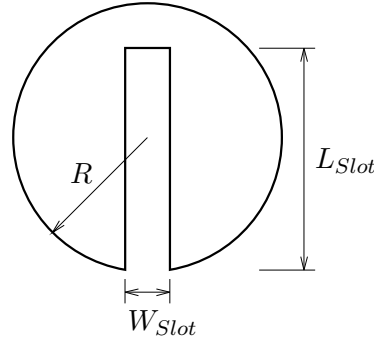


Figure 4.8 Rigid body rotation of a slotted disk: Geometry of the interface

of the interface, the interface is considered as a set of the connected pieces including a circular arc, three finite lines making the rectangular slot and four corner points. The LS function corresponding to the entire interface returns at each point the minimum of the values returned by the LS functions corresponding to the distinguished pieces of the interface. The corresponding SDLS function of a circle is defined by the expression 4.8. The distance to an infinite line characterized by the points (x_1, y_1) and (x_2, y_2) , can be expressed as,

$$d_I(x, y) = \frac{|(x_2 - x_1)(y_1 - y) - (y_2 - y_1)(x_1 - x)|}{\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}} \quad (4.10)$$

The LS function corresponding to a corner point can be expressed by considering the point as a circle with the radius zero. Figure 4.9 a sketch of the regions, within each of which the LS function corresponding to the entire interface returns the minimum distance to a certain piece of the interface. The expression (4.10) returns the distance to an infinite line, not to a piece of the line. Therefore, in order to define the LS function corresponding to a finite line, the domain of definition should be limited. Otherwise,

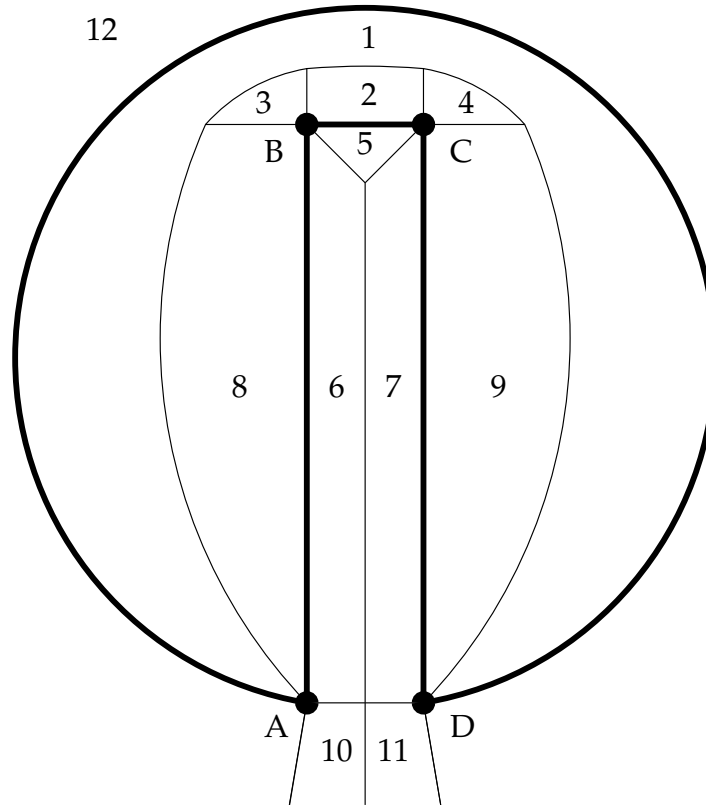


Figure 4.9 Rigid body rotation of a slotted disk: The sub-domains, within each of which the minimum distance to a certain piece of the interface is returned. The picture is a schematic sketch.

in a region such as the sub-domain 3 where the minimum distance to the entire interface is the distance to the point B, the distance to the extension of the finite line AB is returned by mistake. This is in fact the main challenge in constructing the level set function in the present test case. The sub-domain 1 is the region where the minimum distance to the arc AD is returned. The curve separating this region from the sub-domains 2, 3, 4, 8 and 9, is of course determined automatically. Similarly, the curve separating the sub-domains 5, 6 and 7, can also be determined automatically. The sub-domains 10 and 11 are the regions where the minimum distances to the points A and B are returned, respectively. The curve separating these two regions can also be determined automatically. As the arc AD is ended at the points A and D, the sub-domains 10 and 11 are the regions where the minimum distances to these two points are returned, respectively. Figure 4.10 shows the DG field of the constructed SDLS function as well as its zero iso-value as the interface. Moreover, the iso-values of 1.5 and -1.5 are included in the picture, signifying the successful construction of the SDLS function especially in the sub-domains 3, 4, 10 and 11. The prescribed velocity field is defined by the expression 4.9. In order to compare the results with a number of the available calculations, the simulation is performed until the time 628 corresponding to one revolution of the interface.

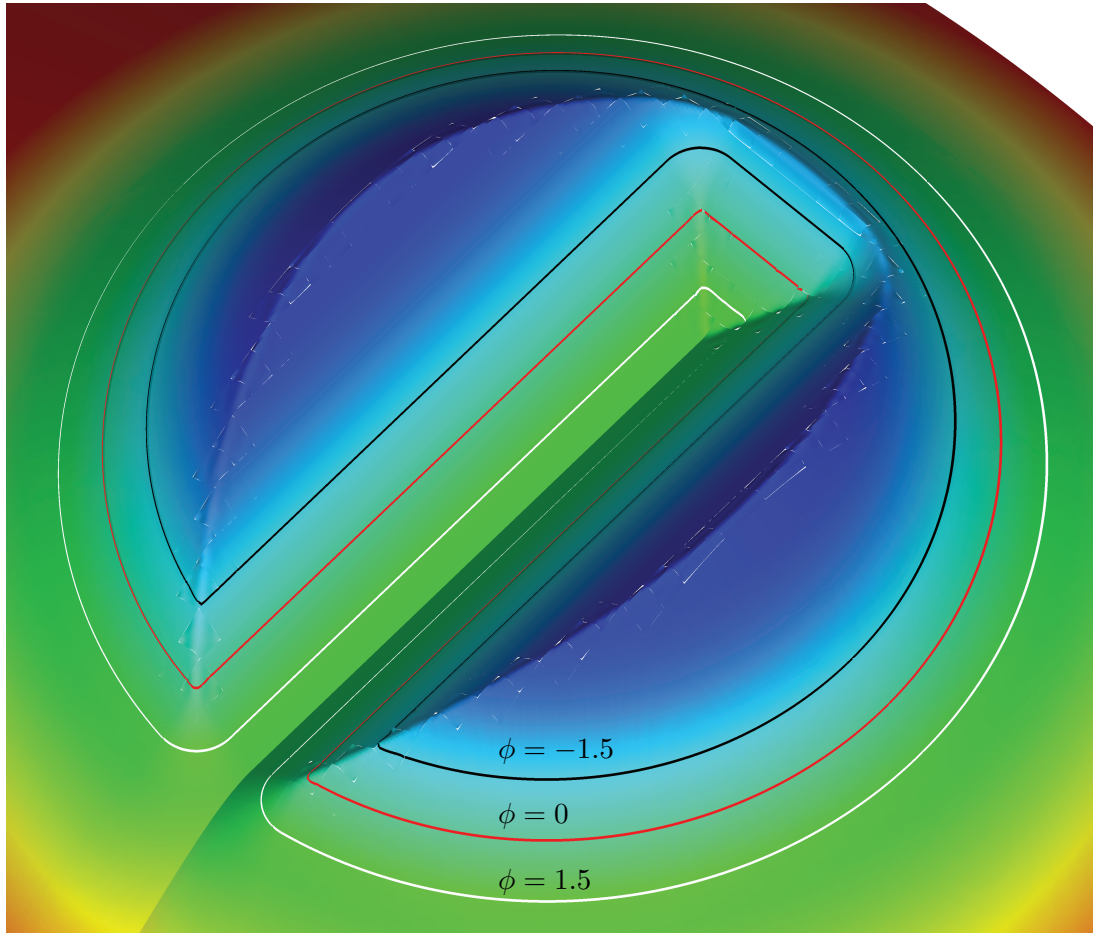


Figure 4.10 Rigid body rotation of a slotted disk: Signed distance LS function of the interface. The OBPS degree $p = 5$ is used together with $N_C = 100 \times 100$.

Numerical Settings The domain is discretized to a set of the quadrilateral cells with $N_C = 25 \times 25$, 50×50 , 100×100 , according to which, the slot passes 2.5 cells, 5 cells and 10 cells along its width, respectively. The OBPS degree for all of the variables is set to $p = 5$. The time step is set to $\Delta t = 0.08$. For each of the variables, a homogeneous Neumann boundary condition is imposed on entire the boundary.

Results Figure 4.11 shows the interface shape after one revolution using different grid resolutions. As it is shown in the pictures, the corners of the slot are rounded because of the dissipation error as well as the finite OBPS degree and the grid resolution. Table 4.2 lists the area loss as well as the interface L^1 -error after one revolution of the interface. In addition to the results obtained in the present research, this table contains the results obtained by Enright et al. (2004) using the LS method and PLS method applying a 5th-order WENO FV method. As it is stated in the table, the accuracy of the DG method using the OBPS degree $p = 5$ together with $N_C = 25 \times 25$ is much higher than the FV method using a 5th-order WENO flux reconstruction scheme together with $N_C = 100 \times 100$.

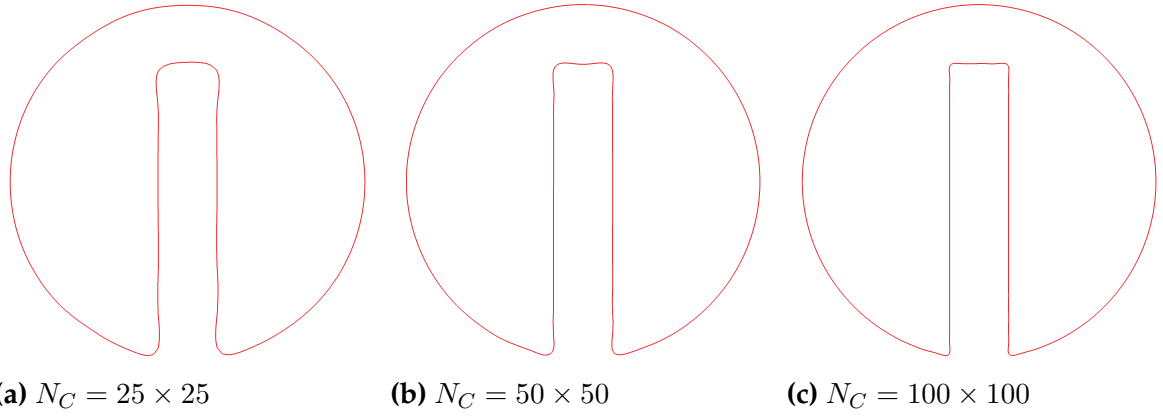


Figure 4.11 Rigid body rotation of a slotted disk: Shape of the interface after one revolution. The OBPS degree is $p = 5$.

Table 4.2

Rigid body rotation of a slotted disk: Accuracy of the DG method comparing to a higher-order WENO FV method

Method	p	N_C	N_{DoF}	Area Loss (%)	\mathbb{L}^1 -error
DG ¹	5	25×25	13125	0.0881	0.0429
DG ¹	5	50×50	52500	0.0312	0.0318
DG ¹	5	100×100	210000	0.00268	0.0153
5 th -Order WENO FV ²	0	100×100	10000	-5.3	0.61
5 th -Order WENO FV ³	0	100×100	10000	0.31	0.07

¹ Present. ² Classical LS method by Enright et al. (2004). ³ PLS method by Enright et al. (2004)

4.3.3 Periodic Swirling of a Circle

This section is assigned to verify the numerical solution to the LSA equation (2.6) by simulating the periodic deformation of an eccentric circle in a prescribed velocity field corresponding to a time-dependent swirling flow. This test case which was originally considered by LeVeque (1996), has been adopted as a standard benchmark by the researchers. The complex geometry of the deformed interface intensifies the effect of the dissipation error. Moreover, the dispersion error takes a major role as a result of the complexity in the temporal and spatial variations of the velocity field. The gradient of the LS function is singular at the center of the circle. The singularity region in this test case is expanded over the domain as a result of the strong deformation of the LS function. Therefore, this test case provides a context for measuring the negative effects of such a singularity on the accuracy of the LS method.

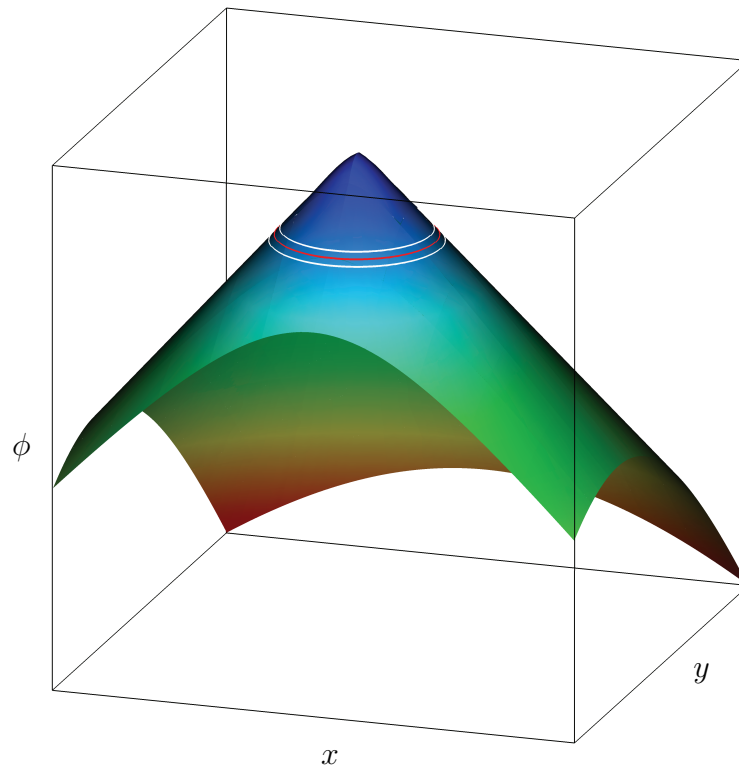
Problem Description The domain of computation is a square with the lower-left corner located at $(0, 0)$ and the upper-right corner located at $(1, 1)$. The initial geometry of the interface is a circle with the radius $R = 0.15$, centered at $(x_c = 0.5, y_c = 0.75)$. The corresponding SDLS function of the interface is defined by the expression (4.8). The gradient of this LS function is singular at the center of the circle. In order to investigate the effects of such a singularity, a non-signed distance level set (NSDLS) function is considered additionally, which does not have a singular gradient. This NSDLS function is defined as,

$$\phi^0(\mathbf{x}) = (x - x_c)^2 + (y - y_c)^2 - R^2. \quad (4.11)$$

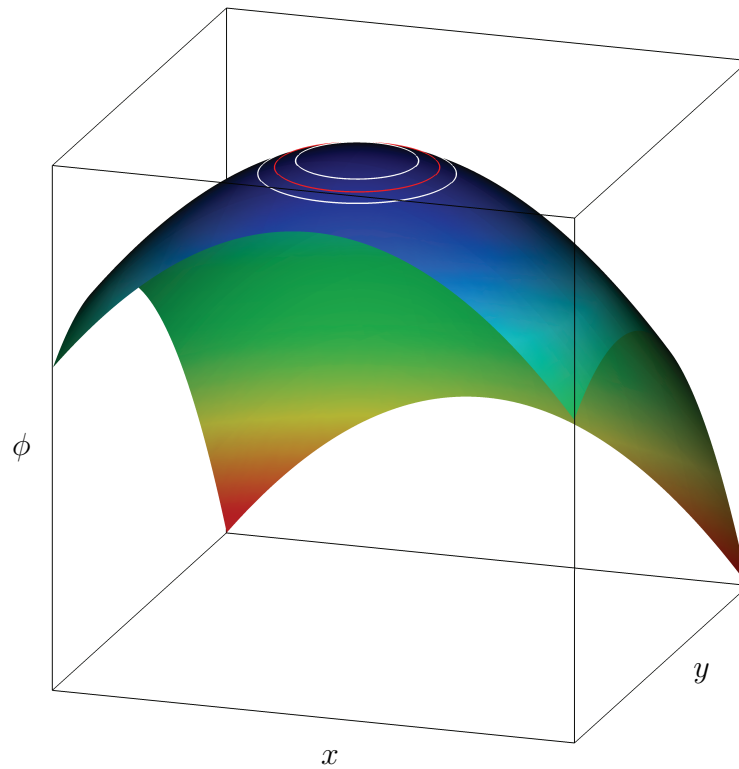
Figures 4.12 shows the difference between the initial SDLS and NSDLS functions of a circle. As it is shown in figure 4.12b, the NSDLS function does not have a sharp apex. Therefore, this function can be properly projected to an OBPS of a certain degree. Figure 4.13 shows the difference between the broken gradients of the SDLS and NSDLS functions after one time step. Figure 4.13a illustrates that although the gradient of the SDLS function is singular only at the center of the circle, the effect of the singularity is not limited to that point. Therefore, in order to reduce the size of the singularity region, one can either make a grid refinement around the original region of the singularity, or using an OBPS of lower-degree in that region. The prescribed velocity field corresponding to a time-dependent swirling flow, is defined as

$$\begin{aligned} \mathbf{u}(\mathbf{x}) &= u_x(x, y)\mathbf{e}_x + u_y(x, y)\mathbf{e}_y \\ &= \sin(2\pi y) \sin^2(\pi x) \cos\left(\frac{\pi t}{T}\right) \mathbf{e}_x \\ &\quad - \sin(2\pi x) \sin^2(\pi y) \cos\left(\frac{\pi t}{T}\right) \mathbf{e}_y \end{aligned} \quad (4.12)$$

where the term $\cos(\pi t/T)$ has been multiplied for making a time periodicity in the interval T . The value of T in this test case is set to 8.

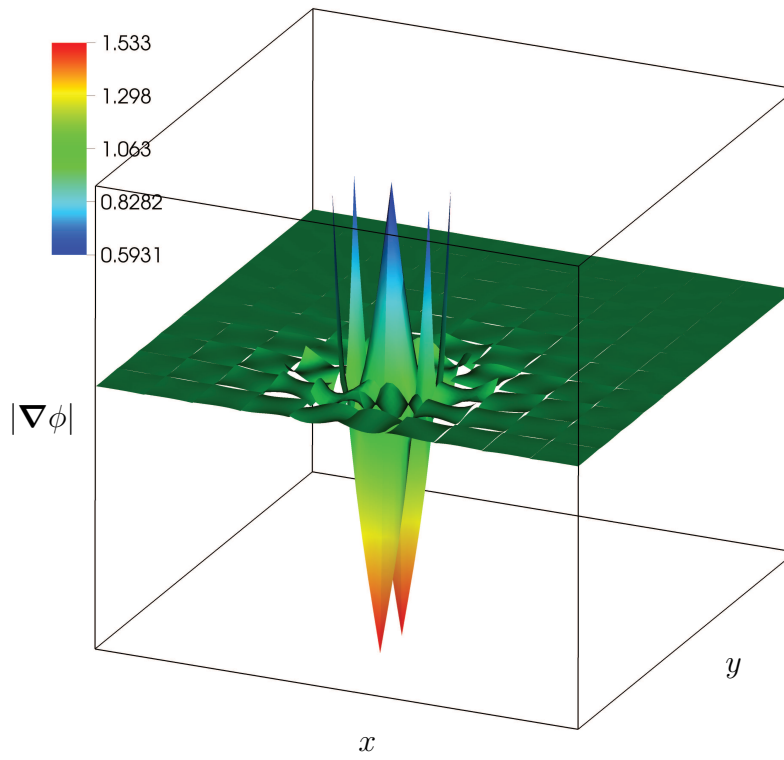


(a) SDLS function

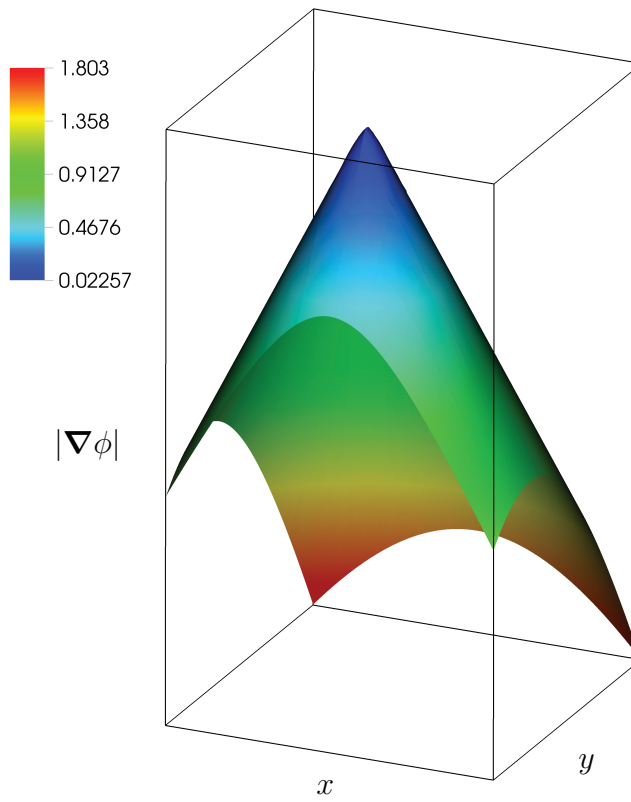


(b) NSDLS function

Figure 4.12 Periodic Swirling of a Circle: SDLS and NSDLS functions of a circle. The OBPS degree $p = 3$ is used with $N_C = 10 \times 10$. The red curve represents $\phi = 0$ and the white curves represent $\phi = -0.01$ and $\phi = 0.01$, respectively.



(a) SDLS function



(b) NSDLS function

Figure 4.13 Periodic Swirling of a Circle: Gradients of the SDLS and NSDLS functions of a circle. The OBPS degree $p = 3$ is used with $N_C = 10 \times 10$.

As the velocity field is time dependent, it should be updated at every time step. As it is mentioned in section 3.6, the time integration of the LSA equation (2.6) is performed applying a 3rd-order TVD RK method described in the section 3.3.2.2. This time integration method consists of three stages. Therefore, the solution accuracy as well as the stability can be highly improved if the velocity field (4.12) is updated after performing each stage of this multi-stage time integration method. A vector plot of this velocity field is shown in figure 4.14.

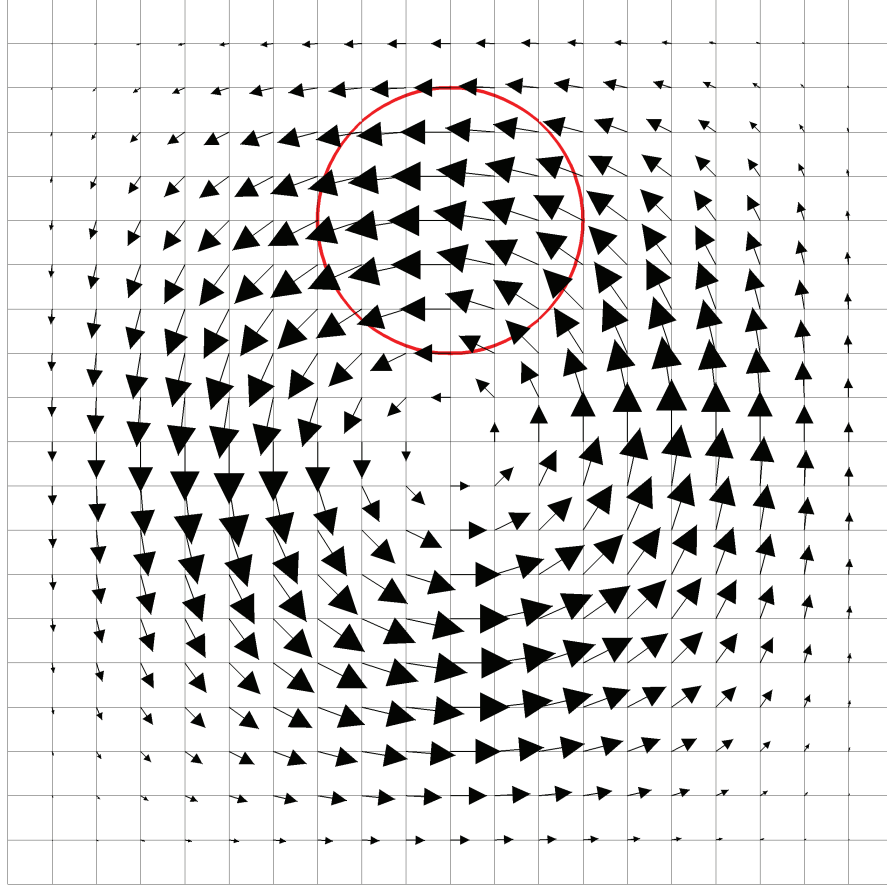


Figure 4.14 Periodic Swirling of a Circle: The prescribed velocity vector field

Numerical Settings This test case is assigned to perform two classes of the convergence studies including the p -convergence study and the h -convergence study, employing both the SDLS and NSDLS functions. In the p -convergence study, the OBPS degrees of $p = 2, 3, 4, 5, 6, 7, 8, 9, 10$ are used and the domain is discretized to a set of the quadrilateral cells with $N_C = 40 \times 40$. The time step for this study is set to 0.00025. In the h -convergence study, the OBPS degrees of $p = 3, 4, 5, 6, 7$ are used and the domain is discretized to the sets of the quadrilateral cells with $N_C = 10 \times 10, 20 \times 20, 40 \times 40, 80 \times 80$. The time step for this study is set to 0.000125. In addition to the convergence studies, a case with $p = 4$ and $N_C = 32 \times 32$ is considered in order to make a comparison with the available results reported in the literature. A homogeneous Neumann boundary condition is imposed on entire the boundary.

Results Figure 4.15 shows a 3D plot the LS function at $t = T/2$ corresponding to its maximum deformation. Figure 4.16 shows a set of the interface shapes captured in one period of the deformation. The results presented in these figures are obtained using $p = 7$ together with $N_C = 160 \times 160$. Moreover, the LS function has initially a non-signed distance property which is defined by the equation (4.11). Hence, these figures show the most accurate result obtained for this test case in the present research.

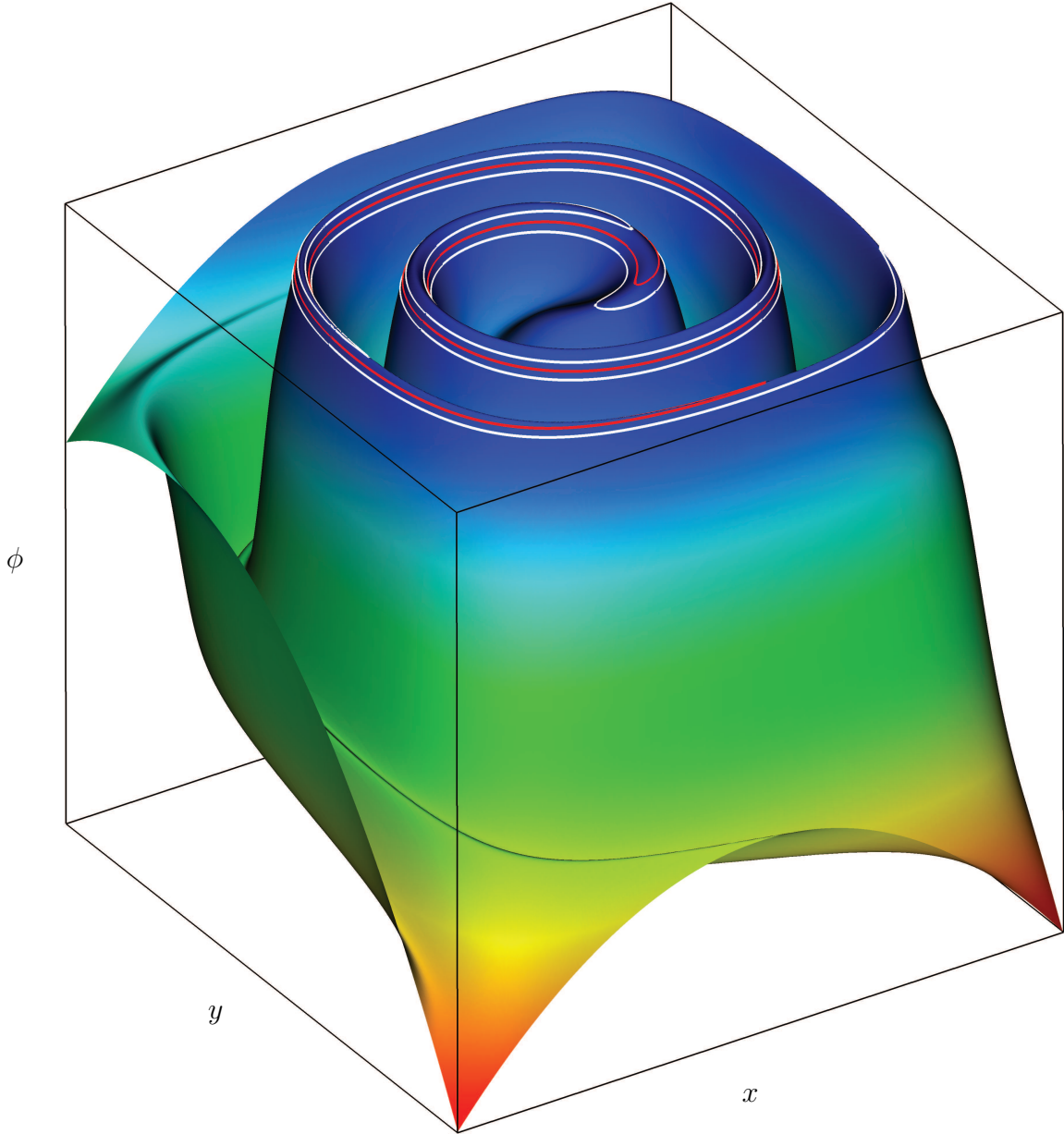


Figure 4.15 Periodic Swirling of a Circle: Level set function at $t = T/2 = 4$ corresponding to its maximum deformation. The OBPS degree $p = 7$ is used together with $N_C = 160 \times 160$. An initial NSDLS function is used defined by equation (4.11). The red curve represents $\phi = 0$ and the white curves represent $\phi = -0.01$ and $\phi = 0.01$.

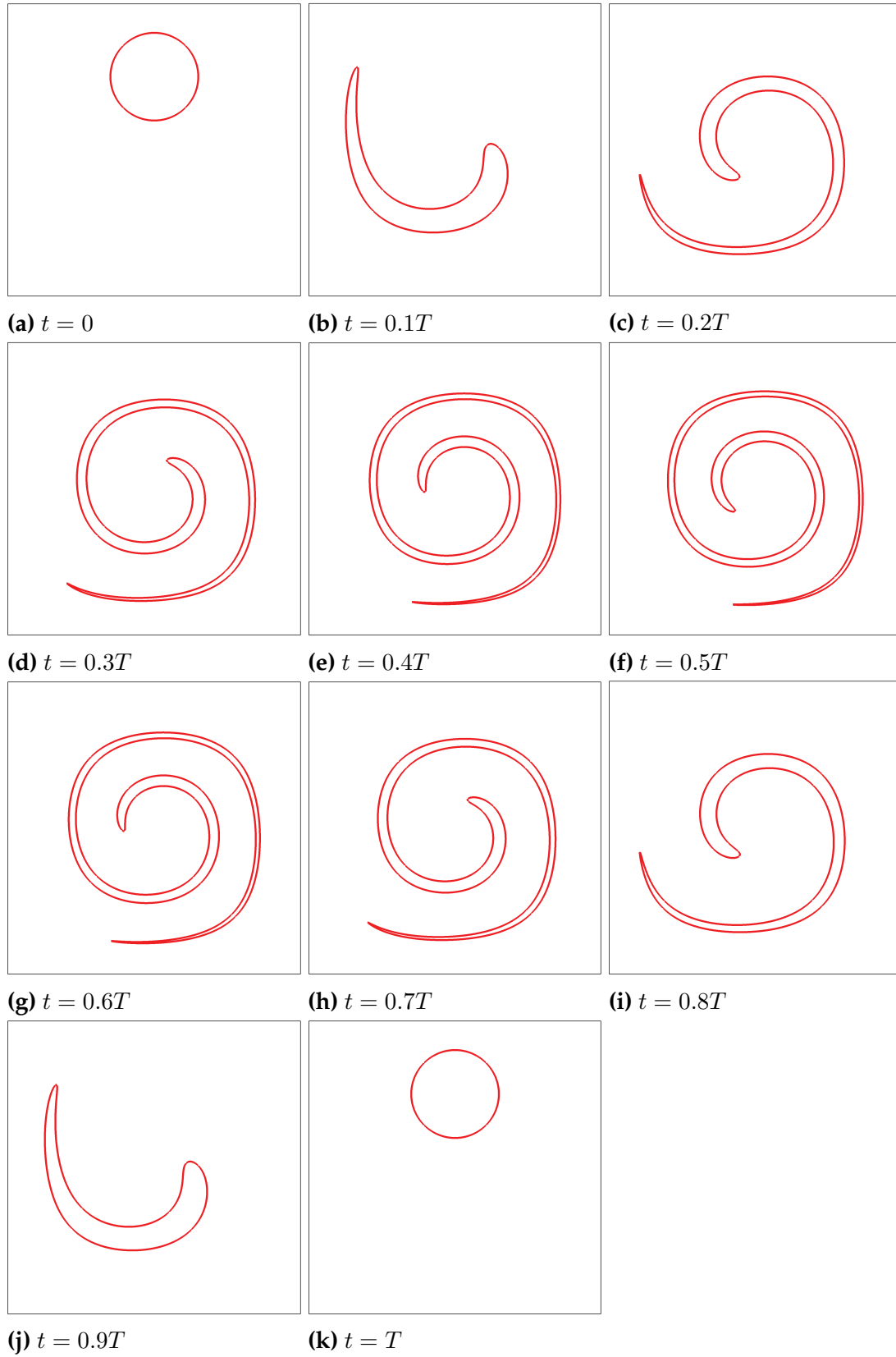


Figure 4.16 Periodic Swirling of a Circle: A completed period of the interface deformation ($T = 8$). The OBPS degree $p = 7$ is used together with $N_C = 160 \times 160$. An initial NSDLS function is used defined by equation (4.11).

Figures 4.17 and 4.18 show the contour plots of the gradients of the initially signed distance and non-signed distance LS functions, using $p = 2, 10$ at $t = T$. The interface is also included in the pictures additionally. These figures are aimed to illustrate the effects of the OBPS degree as well as the singular gradient of the LS function, on the dispersion error of the solution. As it is shown in figures 4.17a and 4.18a, using an OBPS of lower-degree produces a higher amount of the dispersion error regardless of employing an SDLS function or an NSDLS function. Figures 4.17b and 4.18b show that using an OBPS of higher degree dramatically reduces the dispersion error. On the other hand, the spurious waves in the path tracked by the singularity point during the deformation, is quite visible in figure 4.17b, although an OBPS of higher degree is used. It means that the unfavorable effects of the singularity is not limited to the region where the singular point is originally located. Moreover, by looking at figure 4.17b carefully, one can recognize an inaccuracy in regaining the circular shape of the interface at $t = T$.

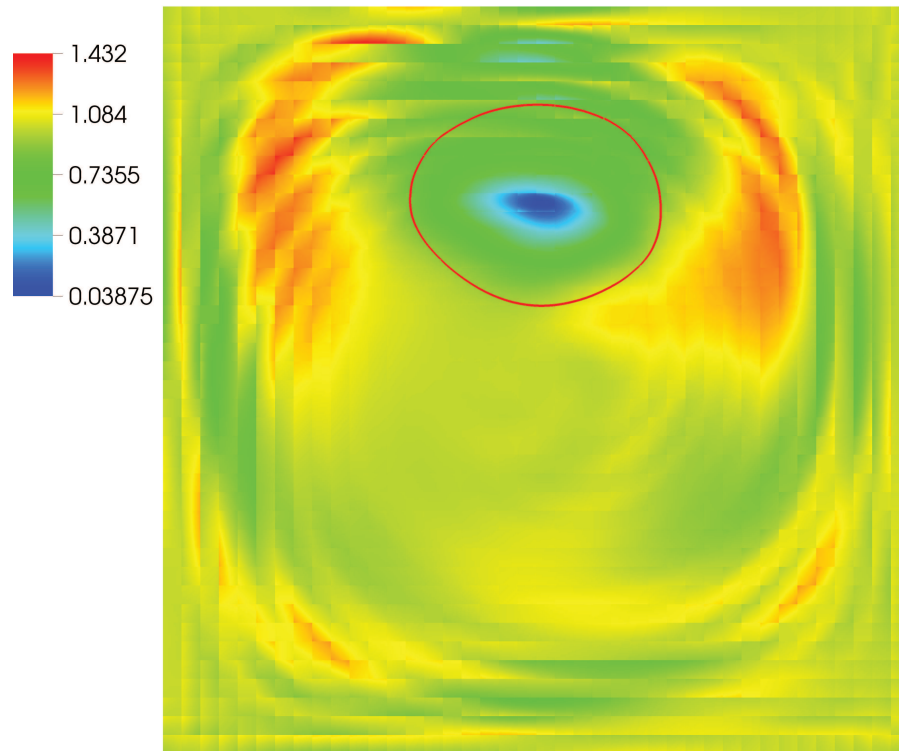
Diagrams 4.19, 4.20 and 4.21 represent the p-Convergence studies on the area error, interface \mathbb{L}^1 -error and LS \mathbb{L}^2 -error at $t = T$, employing both the initially signed distance and initially non-signed distance LS functions. A major fact illustrated in all of these diagrams is that the singularity dramatically reduces the p-convergence rate. As it is explained in section 4.1.2, the interface \mathbb{L}^1 -error is aimed to represent the accuracy in predicting the shape of the interface. Therefore, this error measures the difference between the Heaviside function constructed over the predicted LS function and the one which is constructed over the exact LS function. Whereas, the LS \mathbb{L}^2 -error is aimed to represent the accuracy in predicting the whole LS function including the region where the singularity takes place. Therefore, this error measures the difference between the obtained LS function and the exact one directly. On the other hand, comparing the diagrams 4.20 and 4.21, it is illustrated that the singularity makes almost the same reduction of the p-convergence rate on both of the interface \mathbb{L}^1 -error and the LS \mathbb{L}^2 -error. This supports the statement that the unfavorable effect of the singularity is not limited to the region where the singular point is originally located.

Table 4.3 demonstrates an h -Convergence study on the level set \mathbb{L}^2 -error, employing the initial SDLS function defined by equation (4.8) and initial NSDLS function defined by equation (4.11). This error is theoretically expected to behave as, see e.g. (Pietro & Ern 2011),

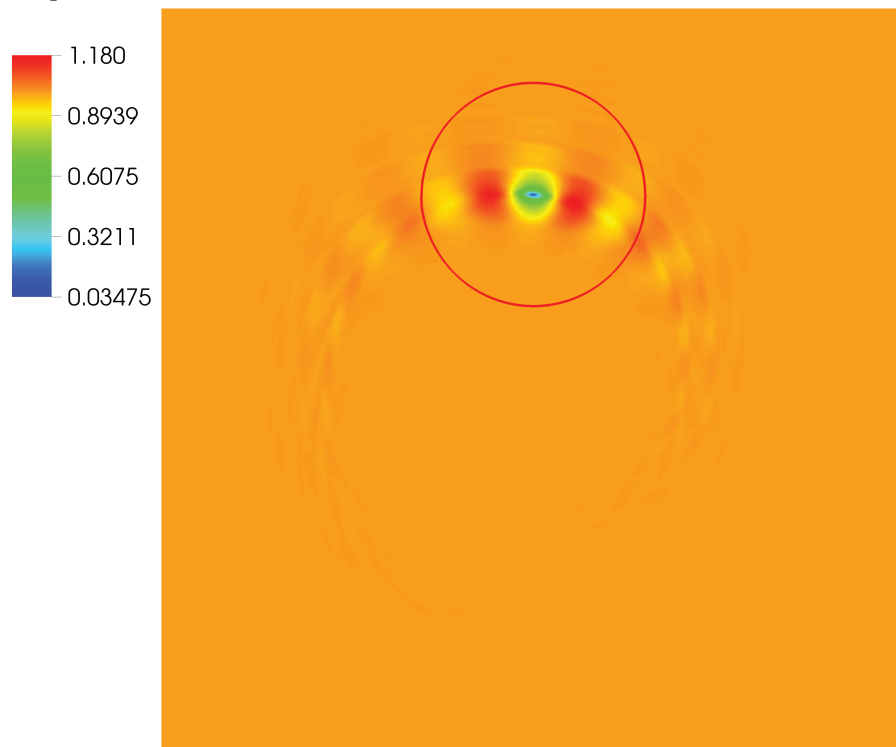
$$\mathbb{L}_\phi^2 = C(p)(h)^{p+1} \quad (4.13)$$

where h denotes the characteristics cell size, p denotes the OBPS degree and C denotes a coefficient which is dependent on p . According to this expression, increasing the grid resolution is theoretically expected to result in an exponential error reduction of order $p + 1$. In table 4.3 the experimental error order (EEO) is calculated for each error with respect to the error corresponding to the coarser grid, as

$$\begin{aligned} \mathbb{L}_{\phi,1}^2 &= C(p)(h_1)^O, \\ \mathbb{L}_{\phi,2}^2 &= C(p)(h_2)^O, \\ O &= \frac{\ln(\mathbb{L}_{\phi,2}^2/\mathbb{L}_{\phi,1}^2)}{\ln(h_2/h_1)}, \end{aligned} \quad (4.14)$$



(a) $p = 2$



(b) $p = 10$

Figure 4.17 Periodic Swirling of a Circle: Gradient of the initially signed distance LS function at $t = T$. The grid resolution is 40×40 .

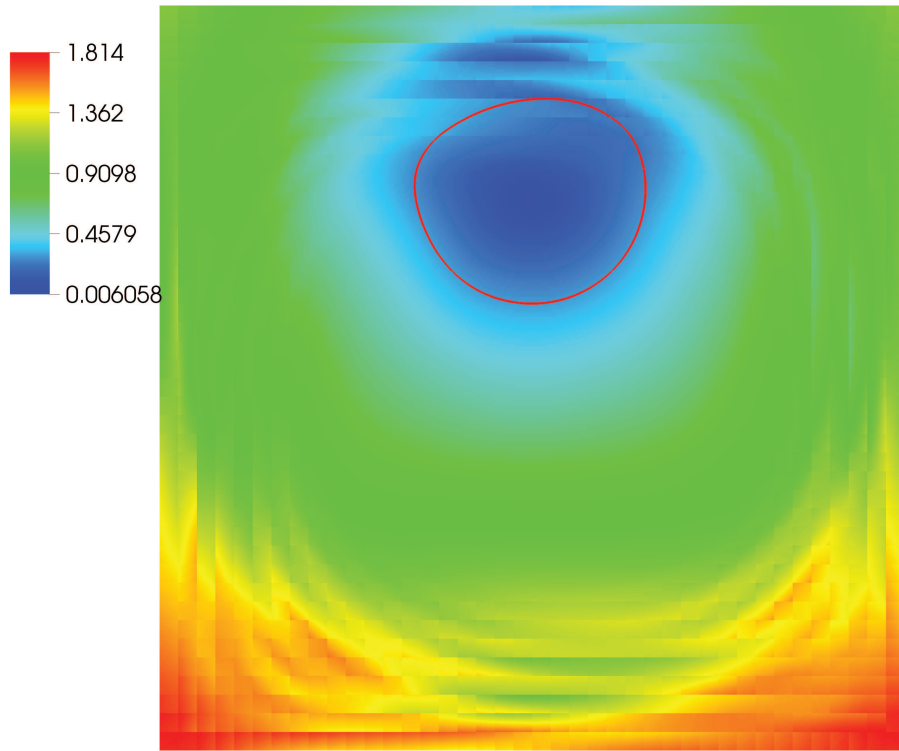
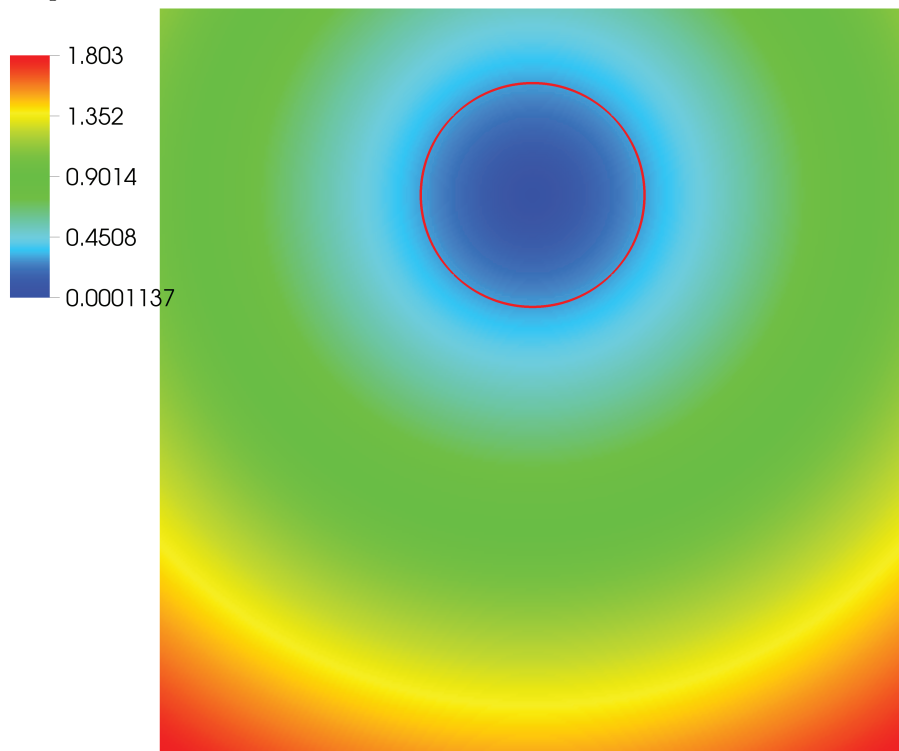
(a) $p = 2$ (b) $p = 10$

Figure 4.18 Periodic Swirling of a Circle: Gradient of the initially non- signed distance LS function at $t = T$. The grid resolution is 40×40 .

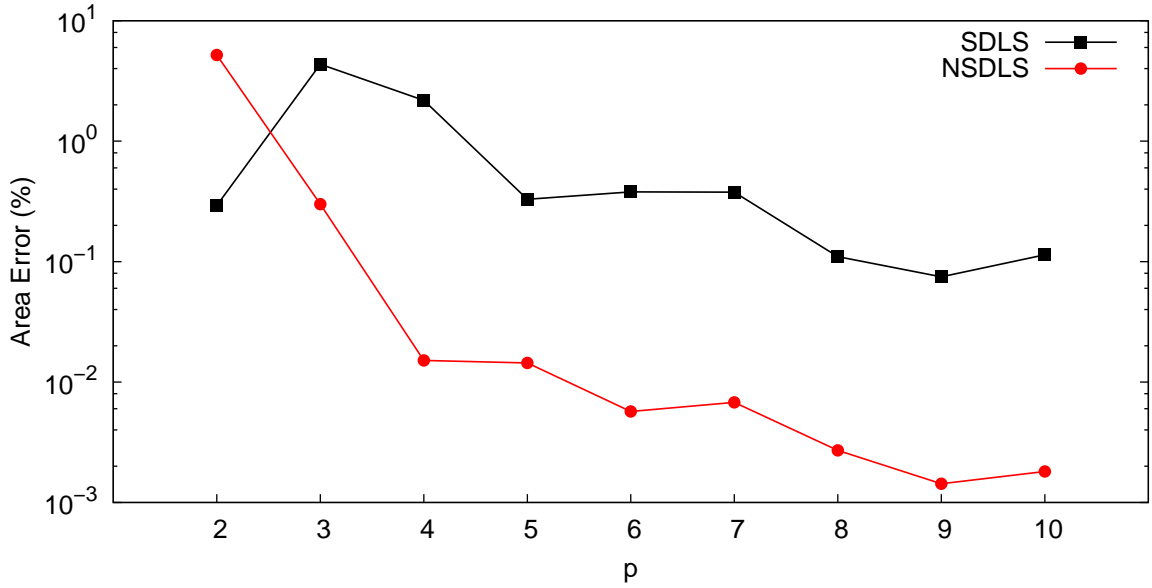


Figure 4.19 Periodic Swirling of a Circle: p-Convergence study on the area error at $t = T$. The grid resolution is $N_C = 40 \times 40$.

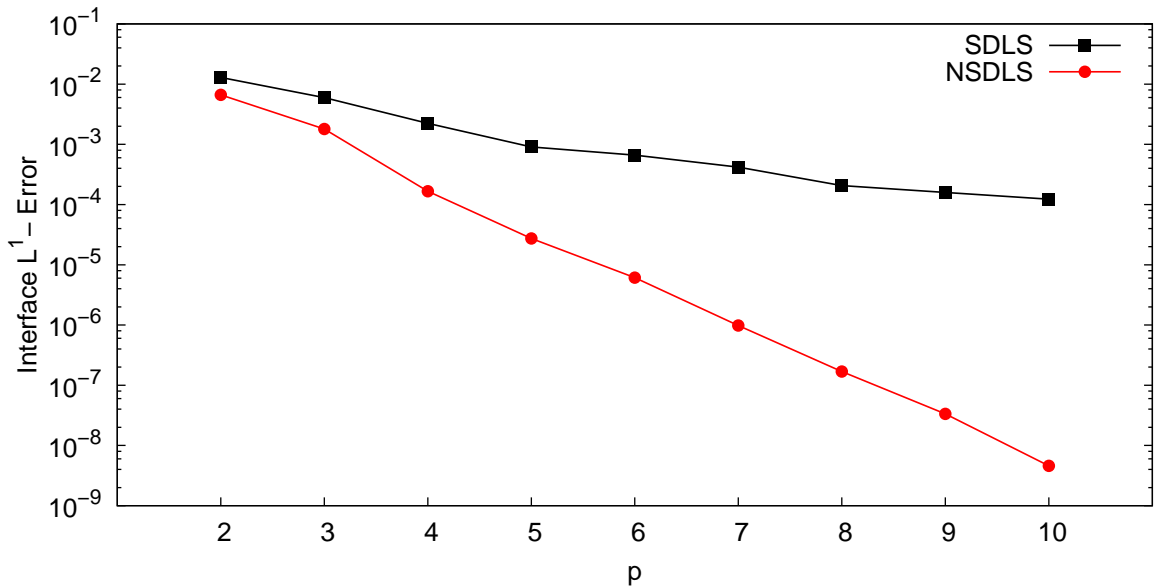


Figure 4.20 Periodic Swirling of a Circle: p-Convergence study on the interface L^1 -error at $t = T$. The grid resolution is $N_C = 40 \times 40$.

where O is the h -convergence rate or EEO. The third column of the table illustrates that employing an initially signed distance LS function does not results in the expected h -convergence rate which is $p + 1$. Rather, it is almost the same for all of the OBPS degrees. According to the fifth column of the table, employing an initially non-signed distance LS function results in the expected h -convergence rate, but not for all of the grid resolutions. It means that using $N_C = 10 \times 10$ and $N_C = 20 \times 20$ is not suitable for this certain problem with the corresponding specifications.

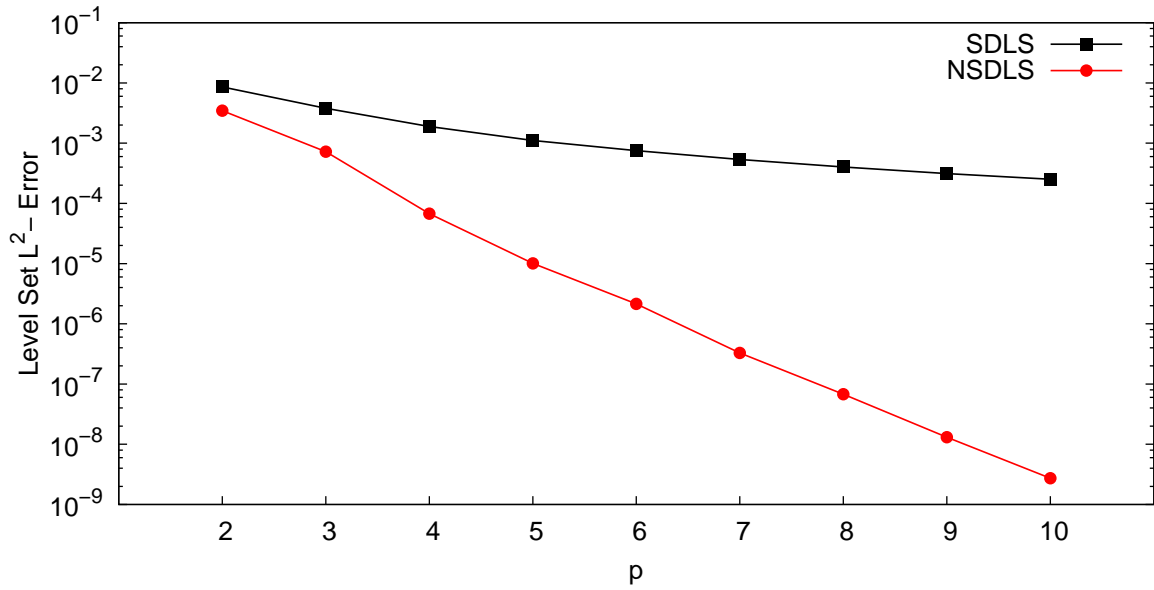


Figure 4.21 Periodic Swirling of a Circle: p -Convergence study on the level set L^2 -error at $t = T$. The grid resolution is $N_C = 40 \times 40$.

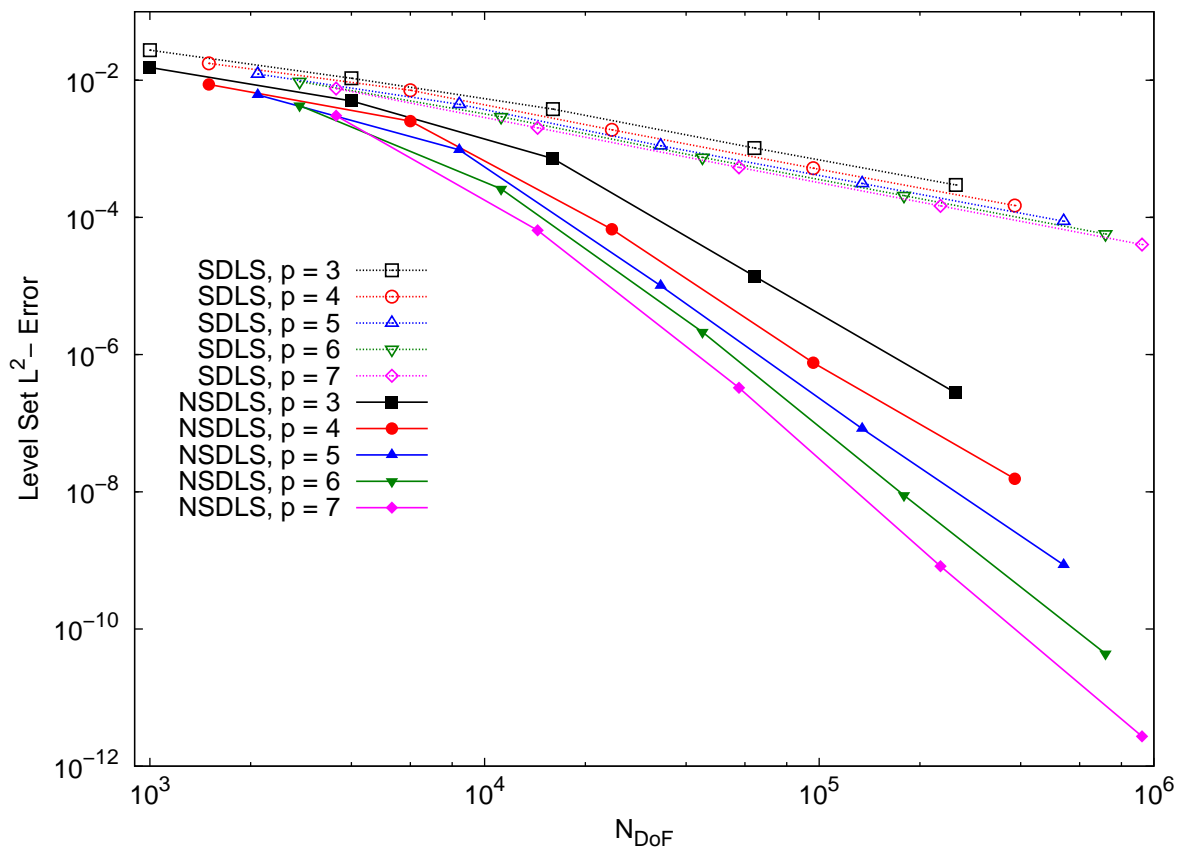


Figure 4.22 Periodic Swirling of a Circle: L^2 -error of LS function versus N_{DoF}

Table 4.3**Periodic Swirling of a Circle: h -Convergence study on the LS \mathbb{L}^2 -error**

p	N_C	Signed Distance		Non- Signed Distance	
		LS \mathbb{L}^2 -Error	EEO	LS \mathbb{L}^2 -Error	EEO
3	10×10	$2.75E - 02$...	$1.54E - 02$...
	20×20	$1.07E - 02$	1.36	$4.99E - 03$	1.63
	40×40	$3.80E - 03$	1.50	$7.20E - 04$	2.79
	80×80	$1.03E - 03$	1.89	$1.39E - 05$	5.69
	160×160	$2.98E - 04$	1.78	$2.76E - 07$	5.66
4	10×10	$1.76E - 02$...	$8.70E - 03$...
	20×20	$7.13E - 03$	1.31	$2.54E - 03$	1.78
	40×40	$1.89E - 03$	1.92	$6.73E - 05$	5.24
	80×80	$5.23E - 04$	1.85	$7.62E - 07$	6.47
	160×160	$1.49E - 04$	1.81	$1.55E - 08$	5.62
5	10×10	$1.23E - 02$...	$6.12E - 03$...
	20×20	$4.45E - 03$	1.47	$9.64E - 04$	2.67
	40×40	$1.11E - 03$	2.00	$1.01E - 05$	6.58
	80×80	$3.12E - 04$	1.83	$8.32E - 08$	6.92
	160×160	$8.79E - 05$	1.83	$8.60E - 10$	6.60
6	10×10	$9.58E - 03$...	$4.28E - 03$...
	20×20	$2.94E - 03$	1.70	$2.61E - 04$	4.04
	40×40	$7.52E - 04$	1.97	$2.14E - 06$	6.93
	80×80	$2.08E - 04$	1.85	$8.91E - 09$	7.91
	160×160	$5.73E - 05$	1.86	$4.36E - 11$	7.67
7	10×10	$7.56E - 03$...	$3.04E - 03$...
	20×20	$2.02E - 03$	1.90	$6.50E - 05$	5.55
	40×40	$5.37E - 04$	1.91	$3.29E - 07$	7.63
	80×80	$1.47E - 04$	1.87	$8.22E - 10$	8.64
	160×160	$4.03E - 05$	1.87	$2.72E - 12$	8.24

Diagram 4.22 is made by plotting the errors listed in table 4.3 versus N_{DoF} . This diagram illustrates that using an OBPS of a higher-degree results an accuracy which is more than the accuracy achieved by using an OBPS of a lower-degree but with the same N_{DoF} . This signifies the computational efficiency achieved by applying the higher-order methods. Table 4.4 makes a comparison between the results obtained in the present research and a number of the available results reported in the literature. As it is illustrated, the accuracy of the DG method is much higher than the FV method with a higher-order WENO flux reconstruction scheme.

Table 4.4

Periodic Swirling of a Circle: Accuracy of the DG method comparing to a higher-order WENO FV method

Method	p	N_C	N_{DoF}	SDLS	Area Loss (%)	\mathbb{L}^1 -error
DG ¹	4	32×32	15360	Yes	-3.81	4.38E-03
DG ¹	4	32×32	15360	No	0.1	7.73E-04
DG ²	4	32×32	15360	No	0.71	5.5E-4
5 th -Order WENO FV ³	0	128×128	16384	Yes	-39.8	3.1E-2
5 th -Order WENO FV ⁴	0	128×128	16384	Yes	-0.71	1.4E-3

¹ Present. ² Quadrature-free DG method by Marchandise et al. (2006). ³ Classical LS method by Enright et al. (2004). ⁴ PLS method by Enright et al. (2004)

4.3.4 Deformation of a Circle in a Channel Flow

This section is assigned to verify the solution to the LSA equation (2.6) by simulating the deformation of a circle in a channel flow corresponding to a fully-developed condition. By considering this test case, it is aimed to investigate the unfavorable effects of imposing a homogeneous Neumann boundary condition at the parts of the boundary where the velocity vector makes an obtuse angle with the normal vector to the boundary. These parts of the boundary are called the inlet.

Problem Description The domain of computation in this test cases is a rectangle with the lower-left corner located at $(0, -0.25)$ and the upper-right corner located at $(2.5, 0.25)$. The initial geometry of the interface is a circle with the radius $R = 0.15$, centered at $(x_c = 0.3, y_c = 0)$. The corresponding SDLS function of the interface is defined by the expression (4.8). The prescribed velocity field is expressed as

$$\mathbf{u}(\mathbf{x}) = u_x(x, y)\mathbf{e}_x + u_y(x, y)\mathbf{e}_y = 1.5U_{avg} \left(1 - \frac{4y^2}{W_{ch}^2}\right) \mathbf{e}_x, \quad (4.15)$$

where W_{ch} is the channel width which is set to 0.5, and U_{avg} is the average velocity which is set to 1.

Numerical Settings The domain is discretized to a set of the quadrilateral cells with $N_C = 50 \times 10$, according to which, the circle passes almost twelve cells along its diameter. The OBPS degree is set to $p = 5$. The time step is set to $\Delta t = 0.00125$. A homogeneous Neumann boundary condition is imposed on the entire boundary. As the left side of the rectangular domain of computation is inlet, this test case is aimed to investigate the effects of using either Neumann or Dirichlet boundary condition on this side.

Results Figure 4.23 shows the 3D plots of the LS function at $t = 0.125, 0.5$, imposing a homogeneous boundary condition on the entire boundary. As it is demonstrated in the figure, the solution undergoes a numerical instability which is slowly developing at the inlet. This is obviously a result of using a homogeneous Neumann boundary condition explained in section 3.4.4. Such an instability does not occur in test cases 4.3.1 and 4.3.2 although some parts of the boundaries are inlet. This is because of the small obtuse angle between the velocity vector and the normal vector to the boundary in these test cases. Figure 4.24 shows the 3D plots of the LS function at $t = 0.125, 0.5$, imposing a Dirichlet boundary condition at the inlet using the local values returned by the initial LS function. A homogeneous Neumann boundary condition is imposed on the rest of the boundary. As it is shown in the figure, the solution is quite stable. Figure 4.25 shows the shape of the interface at $t = 1.25$, corresponding to an area gain of 0.02%.

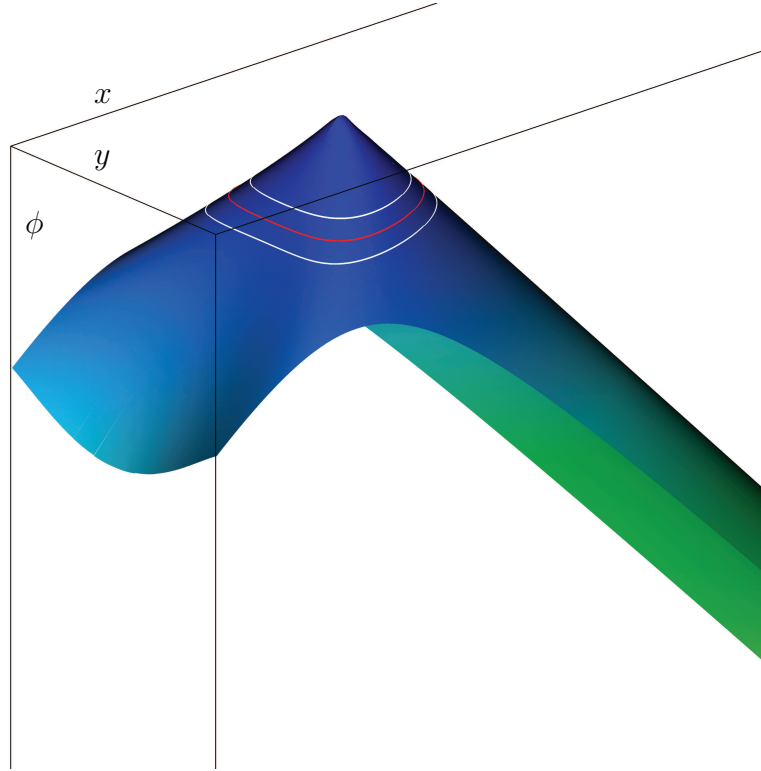
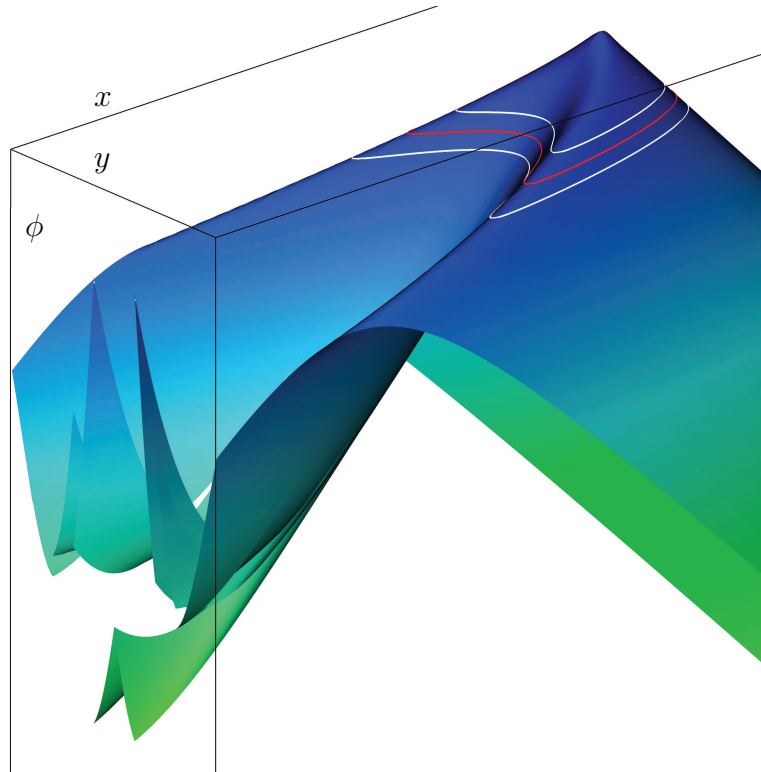
(a) $t = 0.125$ (b) $t = 0.25$

Figure 4.23 Deformation of a Circle in a Channel Flow: 3D plot of the LS function. A Homogeneous Neumann boundary condition is imposed on entire the boundary. The OBPS degree $p = 5$ is used with $N_C = 50 \times 10$. The red curve represents $\phi = 0$ and the white curves represent $\phi = -0.025$ and $\phi = 0.025$.

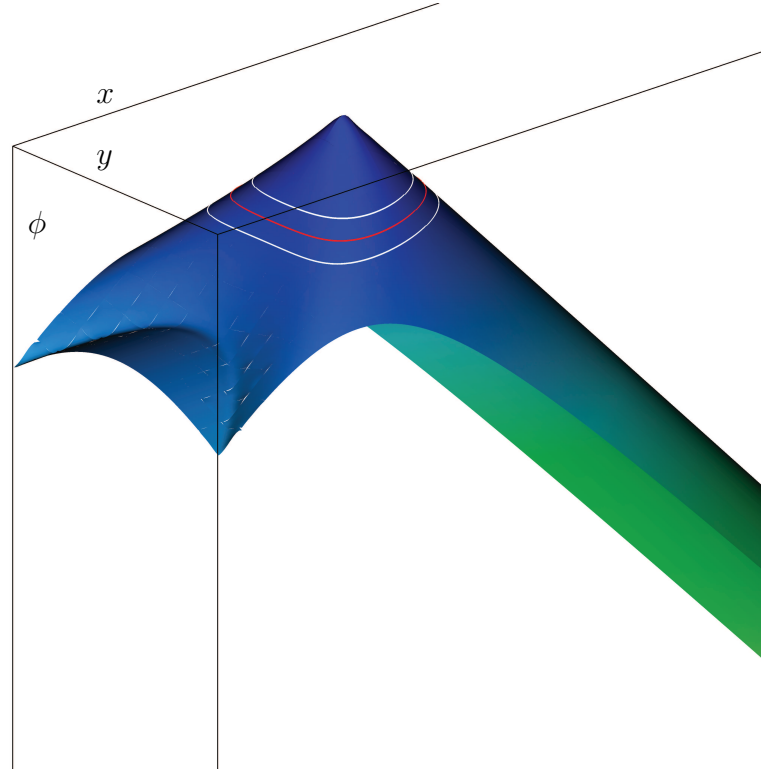
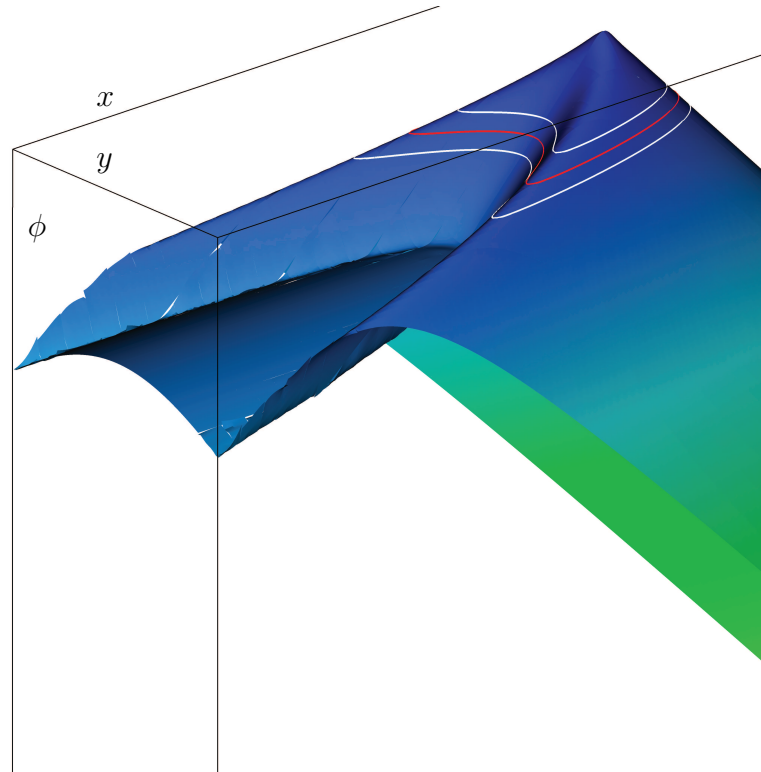
(a) $t = 0.125$ (b) $t = 0.25$

Figure 4.24 Deformation of a Circle in a Channel Flow: 3D plot of the LS function. A Dirichlet boundary condition is imposed on the inlet and a Homogeneous Neumann boundary condition on the rest of the boundary. The OBPS degree $p = 5$ is used with $N_C = 50 \times 10$. The red curve represents $\phi = 0$ and the white curves represent $\phi = -0.025$ and $\phi = 0.025$.

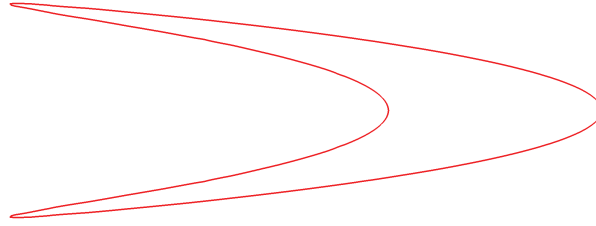


Figure 4.25 Deformation of a Circle in a Channel Flow: Shape of the interface at $t = 1.25$. The OBPS degree $p = 5$ is used with $N_C = 50 \times 10$.

For this certain test case, one can develop also an analytic solution which is described in the Appendix A.

4.3.5 Periodic Swirling of a Slotted Disk

This section is assigned to verify the solution to the LSA equation (2.6) by simulating the periodic deformation of an eccentric slotted disk which is introduced in section 4.3.2, in a prescribed velocity field corresponding to a time-dependent swirling flow which is introduced in section 4.3.3. Hence, this test case is characterized by the complexities in both the interface geometry and the advection field, signifying the domination of both of the dissipation and dispersion errors.

Problem Description The domain of computation is a square with the lower-left corner located at $(0, 0)$ and the upper-right corner located at $(1, 1)$. The initial geometry of the interface is a notched disk with the radius $R = 0.15$, the slot length $L_{Slot} = 0.25$ and the slot width $W_{Slot} = 0.5$, centered at $(x_c = .5, y_c = 0.75)$. The corresponding LS function of the interface can be constructed as it is explained in section 4.3.2. The prescribed velocity field corresponding to a time-dependent swirling flow, is defined by the expression (4.12). The period of deformation is set to $T = 8$.

Numerical Settings The domain is discretized to a set of the quadrilateral cells with $N_C = 100 \times 100$, according to which, the slot passes 10 cells along its width. The OBPS degree is set to $p = 5, 6, 7$. The time step is set to $\Delta t = 0.0002$. A homogeneous Neumann boundary condition is imposed on the entire boundary.

Results Figure 4.26 shows a set of the interface shapes captured in one period of deformation, using $p = 7$. As it is shown in the figure, despite the complexity in the geometry of the interface as well as the long period of deformation, the original shape of the interface is fairly regained at $t = T$. Figure 4.27 shows the shapes of the interface at $t = T$, using different OBPS degrees. Comparing the pictures, one can recognize the difference between the accuracies. Table 4.5 lists the area error as well as the interface L^1 -error at $t = T$, using different OBPS degrees. In this test case, as a result of a certain composition of the area gains and losses at different parts of the interface, the area error produced by using $p = 5$ is lower than the ones produced by using the OBPSs of higher-degrees. But as the interface L^1 -error is based on an absolute difference, it is reduced by increasing the OBPS degree.

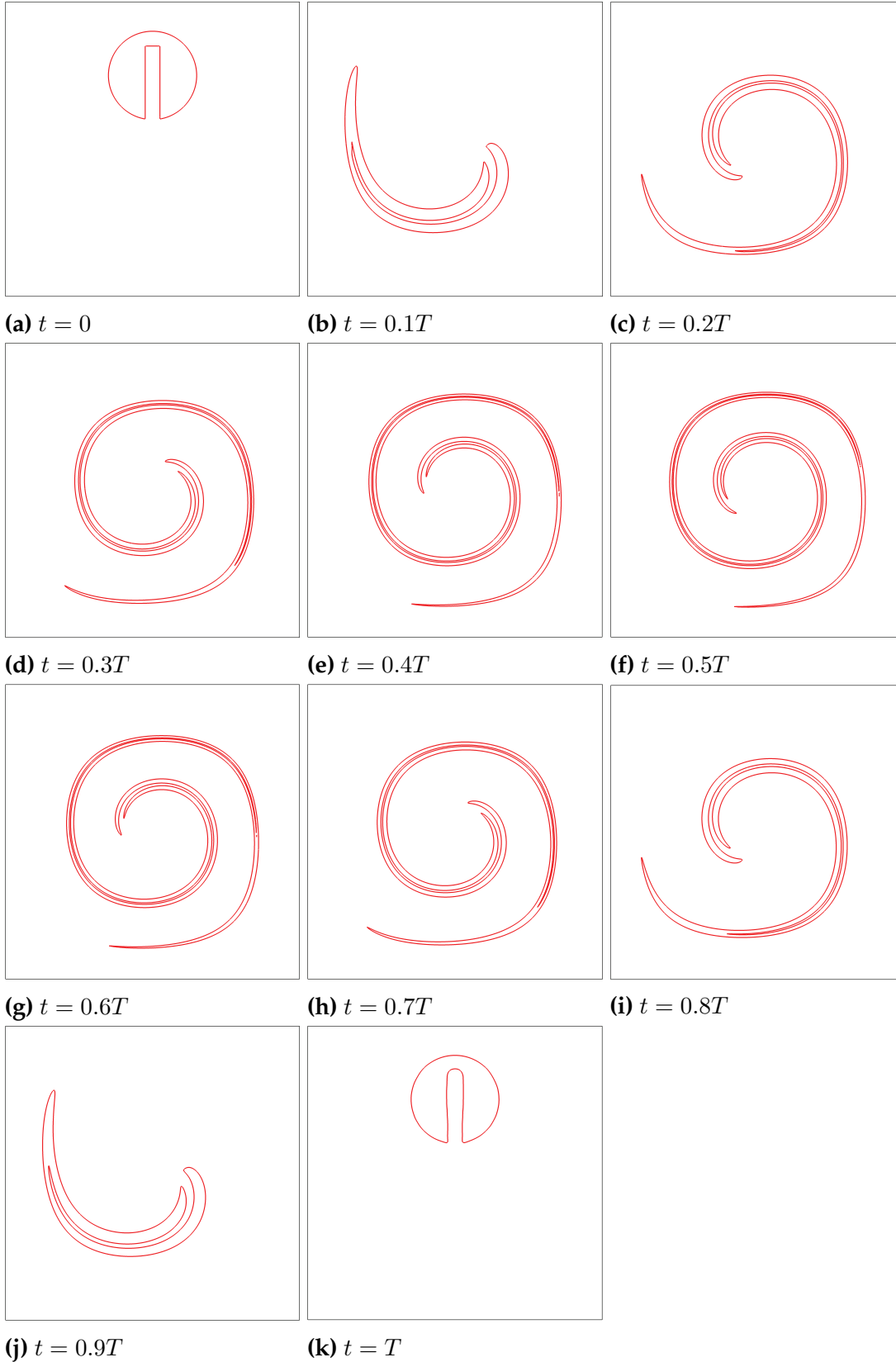


Figure 4.26 Periodic Swirling of a Slotted Disk: A completed period of the interface deformation ($T = 8$). The OBPS degree $p = 7$ is used with $N_C = 100 \times 100$.

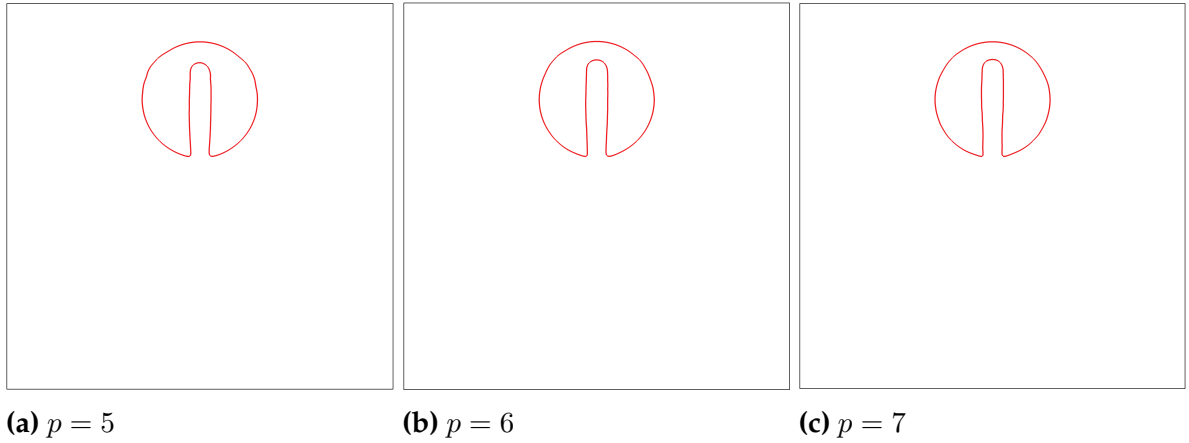


Figure 4.27 Periodic Swirling of a Slotted Disk: Shape of the interface at $t = T$. The grid resolution is $N_C = 100 \times 100$.

Table 4.5

Periodic Swirling of a Slotted Disk: Area loss and the interface L^1 -error at $t = T$. The grid resolution is $N_C = 100 \times 100$.

p	N_{DoF}	Area Loss (%)	L^1 -error
5	210000	-0.332	0.00139
6	280000	-2.0216	0.00121
7	360000	-1.850	0.000974

4.3.6 Deformation of a Circle in a Multi-Vortex Flow

This section is assigned to verify the solution to the LSA equation (2.6) by simulating the deformation of a circle in a prescribed velocity field corresponding to a multi-vortex flow introduced by Smolarkiewicz (1982). This flow field is an artificial flow field which is not necessarily a solution to the NS equation. The multi-rotational characteristics of the velocity field deforms the circle to a very complex geometry. This signify the domination of both of the dissipation and dispersion errors. Although this velocity field is artificial, it can be employed to examine the performance of the DG-based LS method for the cases of advecting interfaces in complex vortical flows such as the turbulent flows.

Problem Description The domain of computation is a square with the lower-left corner located at $(-0.5, -0.5)$ and the upper-right corner located at $(0.5, 0.5)$. The initial geometry of the interface is a circle with the radius $R = 0.15$, centered at $(x_c = 0, y_c = 0)$. The corresponding SDLS function of the initial interface geometry is defined by the expression (4.8). The calculations of this test case was performed before the calculations of the test case 4.3.3, when the advantage of using the NSDLS function defined by the expression 4.11 had not been investigated yet. But it should be generally pointed that constructing an LS function without singularities is only doable for certain simple geometries. The prescribed velocity field is defined as,

$$\begin{aligned} \mathbf{u}(\mathbf{x}) &= u_x(x, y)\mathbf{e}_x + u_y(x, y)\mathbf{e}_y \\ &= -\sin(4\pi(x + 0.5))\sin(4\pi(y + 0.5))\mathbf{e}_x \\ &\quad - \cos(4\pi(x + 0.5))\cos(4\pi(y + 0.5))\mathbf{e}_y \end{aligned} \quad (4.16)$$

This velocity field corresponds to a multi-vortex flow including 16 vortices within the domain of consideration. A vector plot of this velocity field is shown in figure 4.28.

Numerical Settings The domain is discretized to the sets of the quadrilateral cells with $N_C = 20 \times 20, 40 \times 40, 80 \times 80, 160 \times 160, 320 \times 320$. The OBPS degree is set to $p = 10$. The time step is set to $\Delta t = 0.00003125$. As the velocity field is space-periodic in both of the x - and y -directions, the periodic boundary condition is imposed on the domain boundaries.

Results Figure 4.29 shows a 3D plot of the LS function at $t = 0.7$. This plot is made by performing a three-stage cell division for the postprocessing purpose. As it is shown in this figure, that the LS function is highly deformed signifying the necessity of using a higher OBPS degree as well as a higher grid resolution. Figure 4.30 shows a set of the interface shapes captured during the deformation until $t = 0.7$. As it is shown in the figure, the circle is deformed to a complex geometry with very small thickness especially at the borders of the vortical regions. It should be noted that due to a lack of surface tension effects, an interface break up is not supposed to take place.

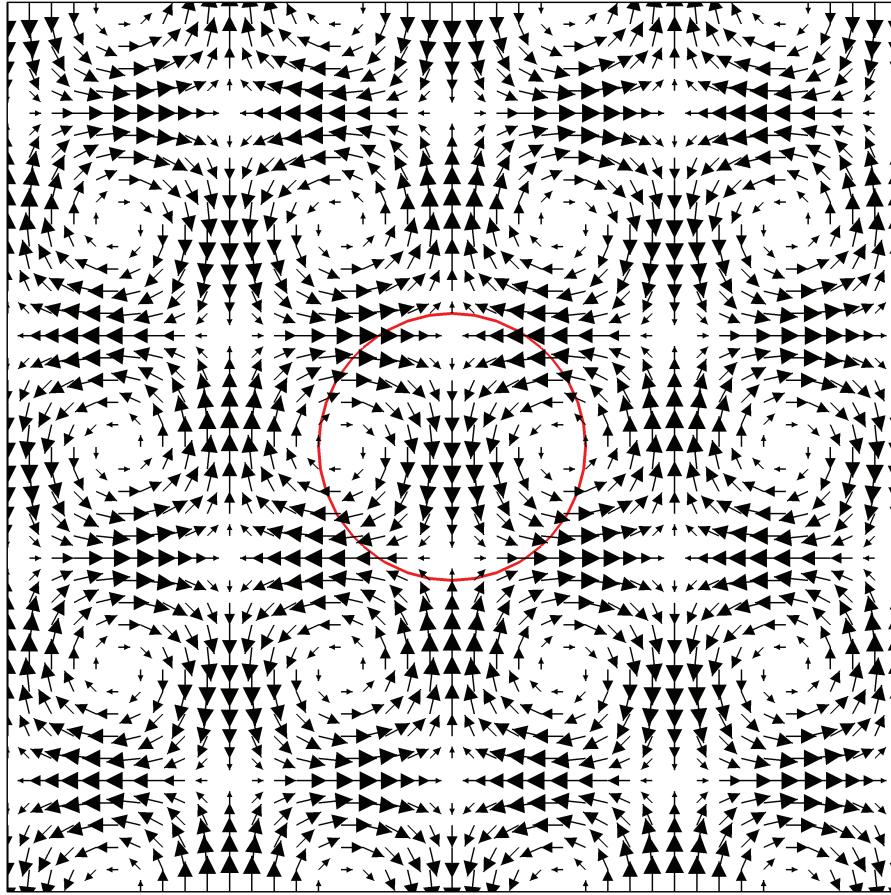


Figure 4.28 Deformation of a Circle in a Multi-Vortex Flow: The prescribed velocity vector field

Figure 4.31 shows a thin part of the deformed object at $t = 0.7$ around the center of the domain. The grid lines shown in the figure are the extra lines added after performing a three-stage cell division for the postprocessing purpose. It means that the original cells are 2^3 times larger than the cells shown in the figure. According to this figure, the thickness of this part of the object is approximately equal to $2 \times 1/16 \times 1/8 \times 1/320 = 0.000048828125$. Table 4.6 lists the area loss at $t = 0.7$ by using different grid resolutions. It is illustrated that as a result of the complex advection field, although an OBPS of degree $p = 10$ is used, still a higher grid resolution is required in order to achieve an acceptable level of the accuracy.

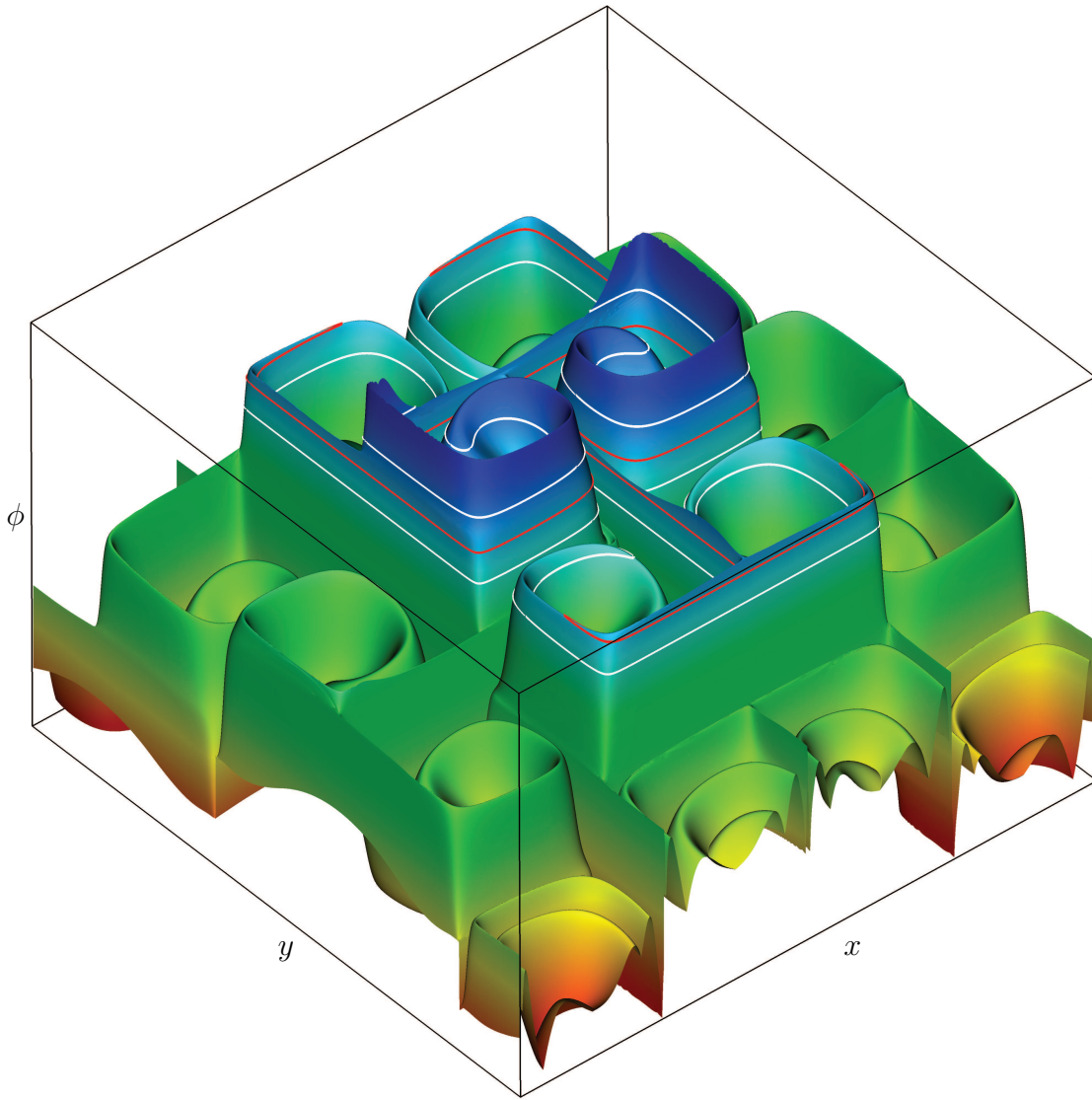


Figure 4.29 Deformation of a Circle in a Multi-Vortex Flow: Level set function at $t = 0.7$. The OBPS degree $p = 10$ is used with $N_C = 320 \times 320$. The initial LS function is signed distance. The red curve represents $\phi = 0$ and the white curves represent $\phi = -0.05$ and $\phi = 0.05$.

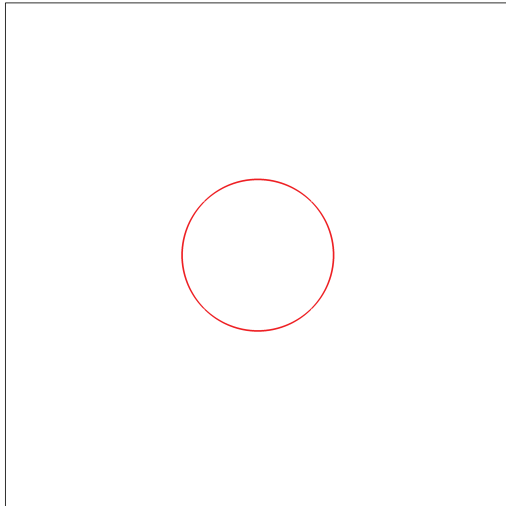
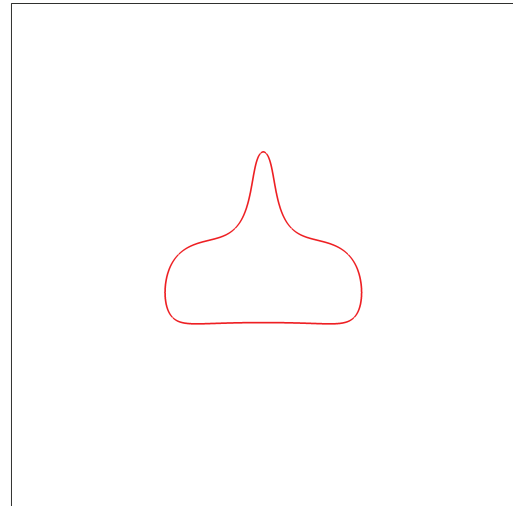
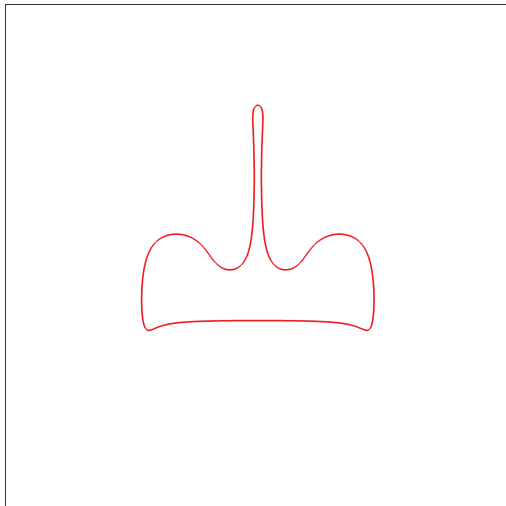
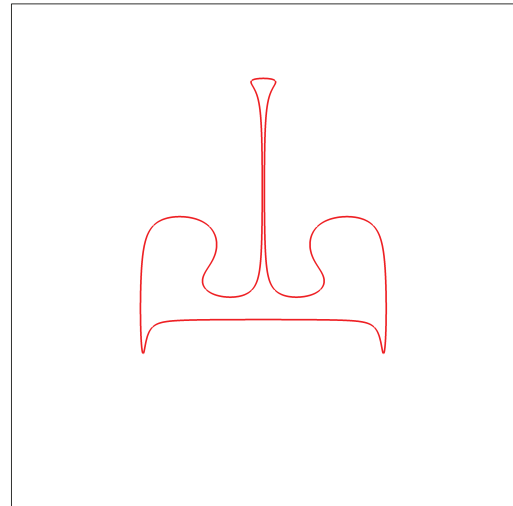
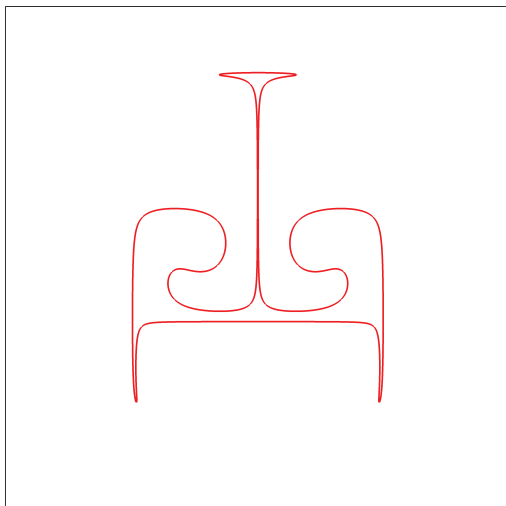
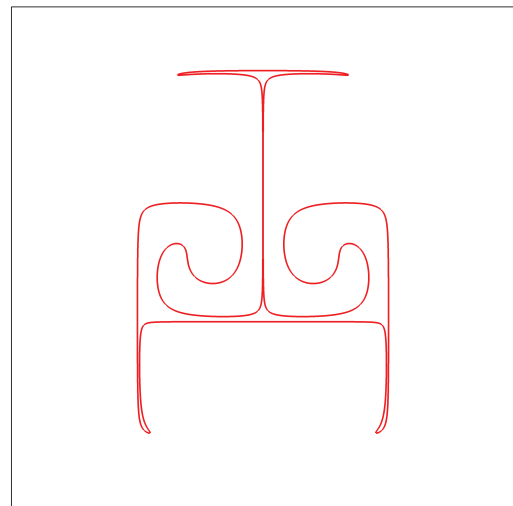
**(a)** $t = 0$ **(b)** $t = 0.1$ **(c)** $t = 0.2$ **(d)** $t = 0.3$ **(e)** $t = 0.4$ **(f)** $t = 0.5$

Figure 4.30 Deformation of a Circle in a Multi-Vortex Flow: Deformation of the interface. The OBPS degree $p = 10$ is used with $N_C = 320 \times 320$. The initial LS function is signed distance.

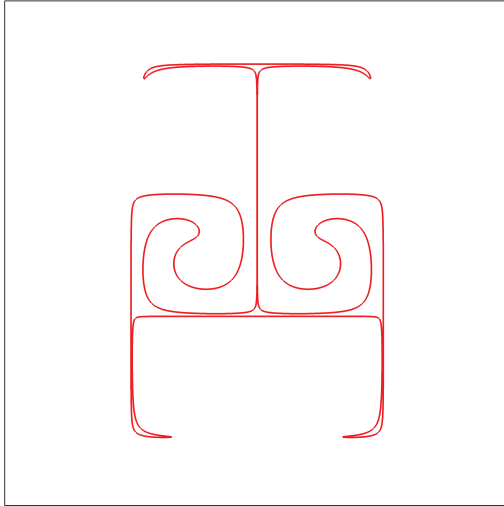
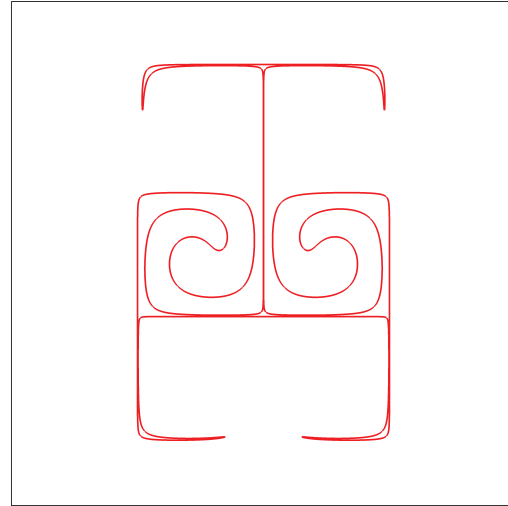
(g) $t = 0.6$ (h) $t = 0.7$

Figure 4.30 Deformation of a Circle in a Multi-Vortex Flow: Deformation of the interface. The OBPS degree $p = 10$ is used with $N_C = 320 \times 320$. The initial LS function is signed distance.

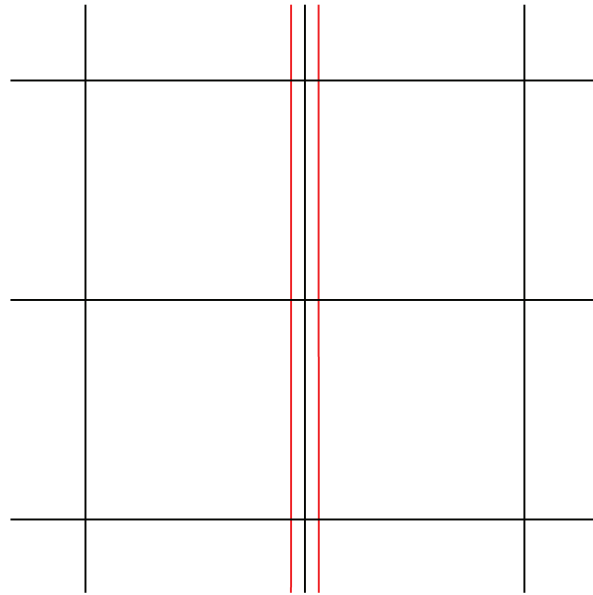


Figure 4.31 Deformation of a Circle in a Multi-Vortex Flow: A part of the deformed geometry around the center of the domain at $t = 0.7$. The OBPS degree $p = 10$ is used with $N_C = 320 \times 320$. The initial LS function is signed distance. The grid lines shown in the picture are the extra lines added after three-stage cell division.

Table 4.6

Deformation of a Circle in a Multi-Vortex Flow: Area loss at $t = 0.7$. The OBPS degree is $p = 10$. The initial LS function is signed distance.

N_C	N_{DoF}	Area Loss (%)
20×20	26400	3.129
40×40	105600	0.556
80×80	422400	0.211
160×160	1689600	0.0600
320×320	6758400	0.0174

4.3.7 Periodic Swirling of a Sphere

This section is assigned to verify the solution to the LSA equation (2.6) by simulating the periodic deformation of an eccentric sphere in a prescribed velocity field corresponding to a time-dependent swirling flow. This test case was originally considered by LeVeque (1996).

Problem Description The domain of computation is a cube with the lower-left-back corner located at $(0, 0, 0)$ and the upper-right-front corner located at $(1, 1, 1)$. The initial geometry of the interface is a sphere with the radius $R = 0.15$, centered at $(x_c = 0.35, y_c = 0.35, z_c = 0.35)$. The corresponding SDLS function of the interface is analytically defined as,

$$\phi^0(\mathbf{x}) = \sqrt{(x - x_c)^2 + (y - y_c)^2 + (z - z_c)^2} - R. \quad (4.17)$$

The calculations of this test case was performed before the calculations of the test case 4.3.3, when the advantage of using an LS function without singularities had not been investigated yet. But it should be generally pointed that constructing an LS function without singularities is only doable for certain simple geometries. The prescribed velocity field is defined as,

$$\begin{aligned} \mathbf{u}(\mathbf{x}) &= u_x(x, y, z)\mathbf{e}_x + u_y(x, y, z)\mathbf{e}_y + u_z(x, y, z)\mathbf{e}_z \\ &= 2\sin^2(\pi x)\sin(2\pi y)\sin(2\pi z)\cos\left(\frac{\pi t}{T}\right)\mathbf{e}_x \\ &\quad - \sin(2\pi x)\sin^2(\pi y)\sin(2\pi z)\cos\left(\frac{\pi t}{T}\right)\mathbf{e}_y \\ &\quad - \sin(2\pi x)\sin(2\pi y)\sin^2(\pi z)\cos\left(\frac{\pi t}{T}\right)\mathbf{e}_z, \end{aligned} \quad (4.18)$$

where the term $\cos(\pi t/T)$ has been multiplied for making a time periodicity in the interval T . The value of T in this test case is set to 3. As the velocity field is time dependent, it should be updated at every time step. As it is mentioned in section 3.6, the time integration of the LSA equation (2.6) is performed applying a 3rd-order TVD RK method described in the section 3.3.2.2. This time integration method consists of three stages. Therefore, the solution accuracy as well as the stability can be highly improved if the velocity field (4.12) is updated after performing each stage of this multi-stage time integration method.

Numerical Settings The domain is discretized to a set of the non-uniformly distributed cuboidal cells with $N_C = 40 \times 40 \times 40$. The OBPS degree is set to $p = 5$. The time step is set to $\Delta t = 0.0005$. A homogeneous Neumann boundary condition is imposed on the domain boundaries.

Results Figure 4.32 shows a set of the interface shapes captured in a completed period of the deformation. Figure 4.33 shows the shape of the interface at $t = 1.4T$ signifying the necessity of using a higher-degree OBPS as well as a higher grid resolution in order to accurately capture the small thickness of the deformed object. Table 4.7 makes a comparison between the results obtained in the present research and a number of the available results reported in the literature.

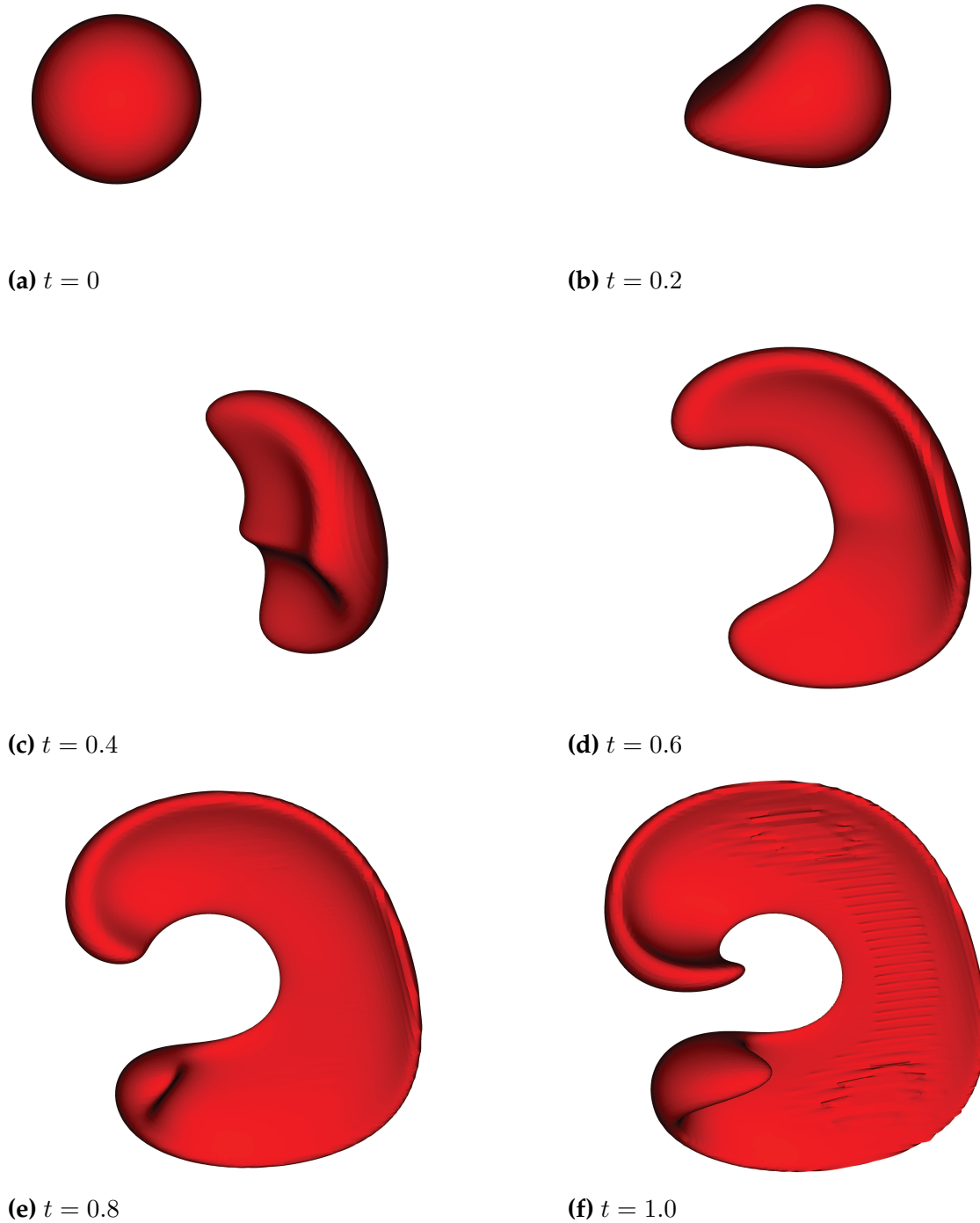


Figure 4.32 *Periodic Swirling of a Sphere:* A completed period of the interface deformation ($T = 3$). The OBPS degree $p = 5$ is used with $N_C = 40 \times 40 \times 40$. The initial LS function is signed distance.

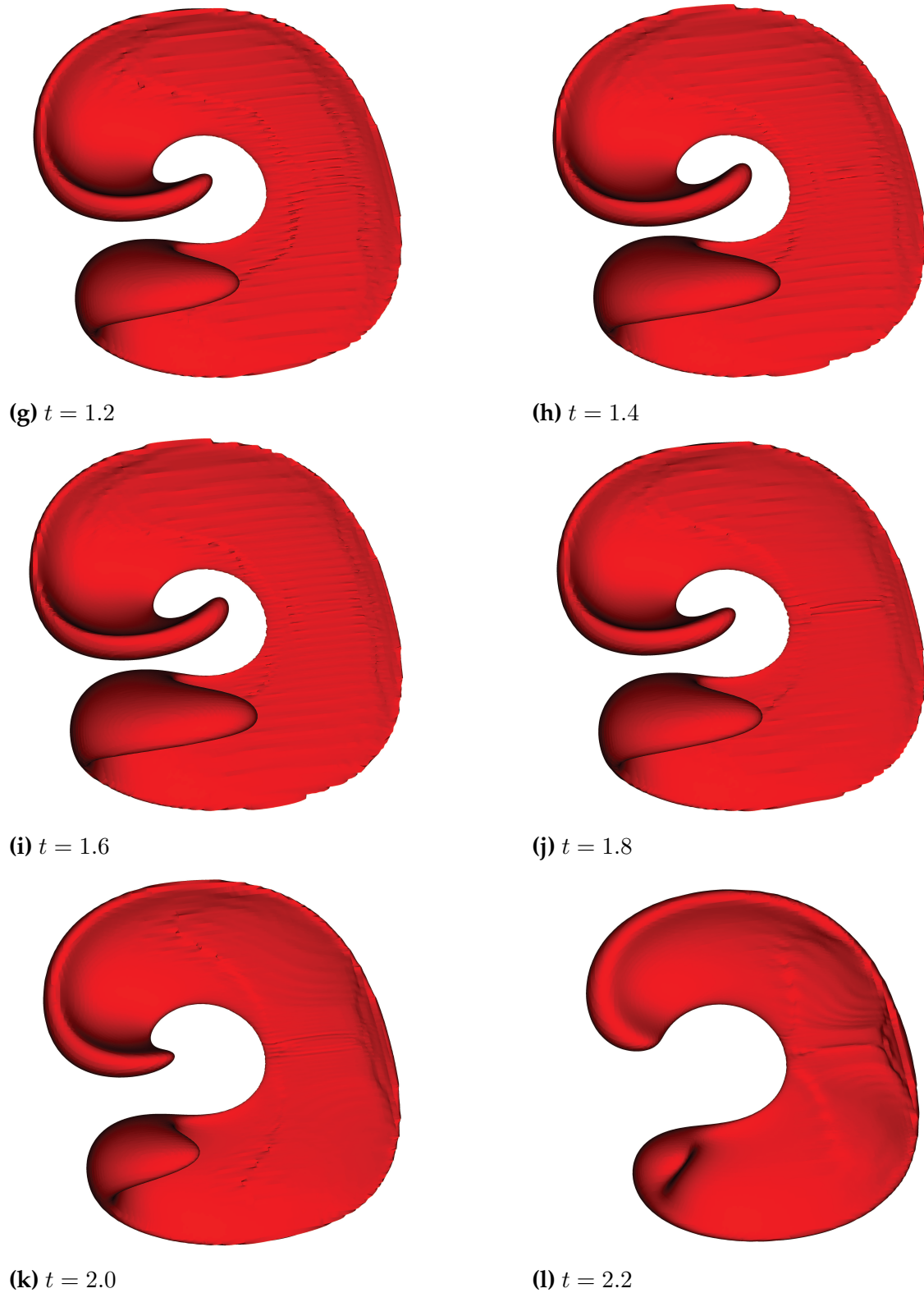


Figure 4.32 *Periodic Swirling of a Sphere:* A completed period of the interface deformation ($T = 3$). The OBPS degree $p = 5$ is used with $N_C = 40 \times 40 \times 40$. The initial LS function is signed distance.

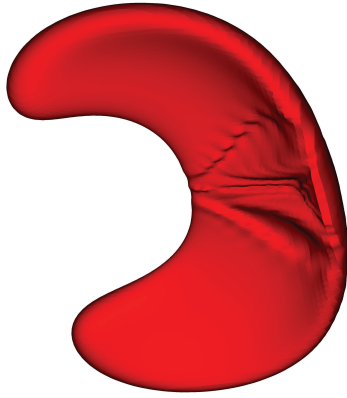
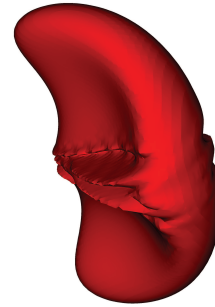
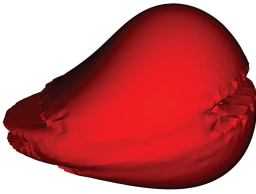
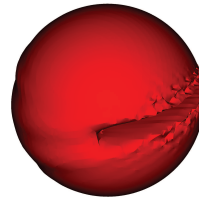
(m) $t = 2.4$ (n) $t = 2.6$ (o) $t = 2.8$ (p) $t = 3.0$

Figure 4.32 *Periodic Swirling of a Sphere:* A completed period of the interface deformation ($T = 3$). The OBPS degree $p = 5$ is used with $N_C = 40 \times 40 \times 40$. The initial LS function is signed distance.

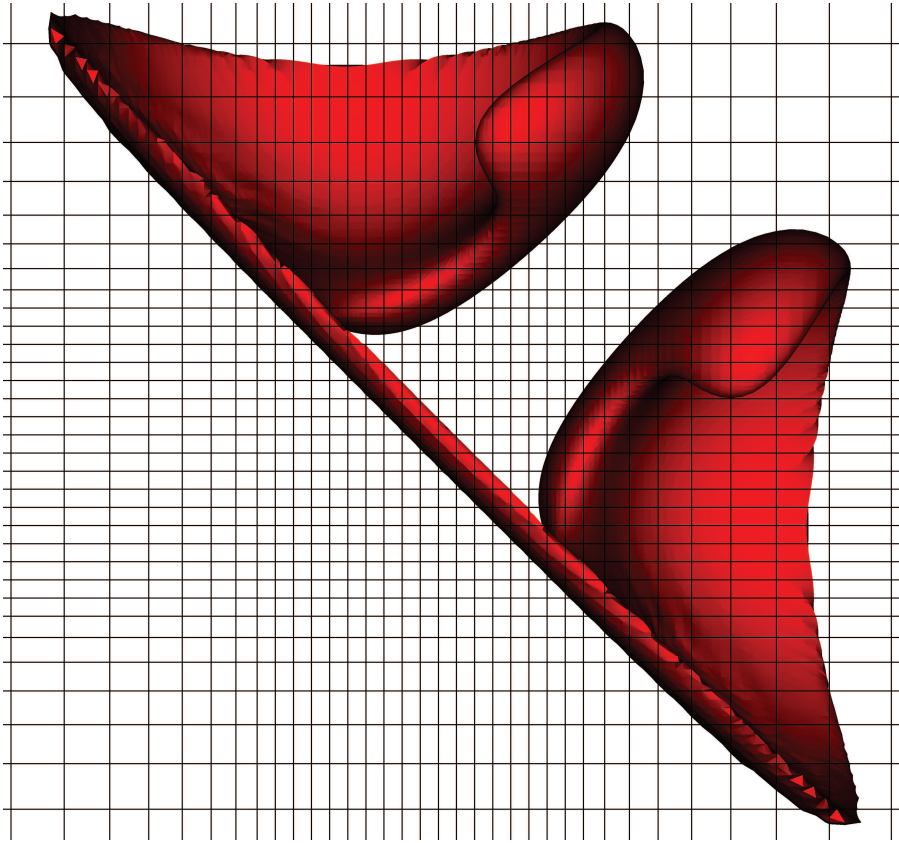


Figure 4.33 *Periodic Swirling of a Sphere:* Shape of the interface at $t = 1.4$. The OBPS degree $p = 5$ is used with $N_C = 40 \times 40 \times 40$. The initial LS function is signed distance.

Table 4.7
Periodic Swirling of a Sphere: Accuracy of the DG method comparing to a higher-order WENO FV method

Method	p	N_C	N_{DoF}	Area Loss (%)	\mathbb{L}^1 -error
DG ¹	5	$40 \times 40 \times 40$	3584000	1.813	0.00227
5 th -Order WENO FV ²	0	$100 \times 100 \times 100$	1000000	80	...
5 th -Order WENO FV ³	0	$100 \times 100 \times 100$	1000000	2.6	...

¹ Present. ² Classical LS method by Enright et al. (2004). ³ PLS method by Enright et al. (2004)